A Tutorial to the Symbolic Computation System Maple

符号计算系统 Maple教程

张韵华　王新茂■编著

中国科学技术大学出版社

目　　录

绪　论

0.1　符号计算与数值计算

在高级语言环境下，数值计算即计算数学表达式的值是计算机的主要工作之一。数学表达式由常量、变量、函数和运算符等组成，最后的计算结果是一个数。一个数值计算问题要经过多步浮点运算，方法的收敛性、收敛速度、舍入误差和误差累计与计算结果密切相关。例如，用C语言编程计算 $z=(x+y)^3-(x^3+3x^2y+3xy^2+y^3)$，其中 $x=10000$，$y=0.0001$，得到结果 $z=-0.0001220708125$。但是，我们知道，对任意的 x，y，表达式 z 的值都应该为0。

"符号"包括"数值"，"符号"远比"数值"表示的范围广。符号计算是研究如何在计算机上表示和处理数学符号，设计进行数学公式的演算和推导的高效计算方法，以及通过计算机程序和软件系统实现这些算法。符号计算包括数值计算和数学推理，数学推理表现在对数学表达式的化简和函数变换上。例如，对多项式因式分解，计算不定积分。符号计算的处理对象是具有含义的数学符号，如整数、数学常数、字母、函数等。在符号计算环境下，表达式仍由常量、变量、函数和运算符等组成，最后的计算结果是经过数学变换的表达式，可以是一个数值、变量、函数，也可以是含有运算符的表达式。对于精确数计算结果总是精确的，没有误差的。例如，在Maple中计算 $z:=(x+y)^3-(x^3+3x^2y+3xy^2+y^3)$，用符号计算方式直接化简Simplify[z]的结果是精确数0；如果输入 $x:=10000$，$y:=1/10000$，则系统做精确数运算得到结果 $z:=0$；如果输入的是 $x:=10000$，$y:=0.0001$，则系统做数值运算得到结果 $z:=-0.000030000001$。又例如：对未赋初值的变量 x，因式分解 factor(x^2-5x+6)的结果是多项式乘积$(x-2)(x-3)$。

符号计算系统是进行符号计算的计算机软件系统。目前一些流行的符号计算系统有：Aldor，Axiom，CoCoA，Derive，GAP，Macaulay，Macsyma，Magma，Maple，Mathematica，Maxima，MuPAD，Reduce，Singular 等。其中既有为特殊目的开发的专用软件系统，也有集成了符号计算、数值计算、图形界面、高级语言编程等功能的通

用软件系统;既有免费的开放源代码软件,也有价值不菲的商业软件。

符号计算系统的对象从初等数学到高等数学,几乎涉及所有数学领域。从表达式的化简、整数分解、导数、积分、方程求解、微分方程、矩阵计算到码的构造、环的理想、群的表示、范畴的定义等。

符号计算系统已被成功地应用于绝大多数的科学研究和工程技术领域,它也是数学领域和数学定理机械化证明的有力工具。由于它能够正确地完成人脑在短时间内无法完成的公式推导计算,应用符号计算系统,使得不少研究领域的前沿得到了进一步的推进。

0.2 Maple 简介

Maple 是目前世界上最流行的通用数学软件系统之一。最初是作为加拿大 Waterloo大学和瑞士 ETH Zurich 大学的一个合作科研项目,于 1980 年由 Keith Geddes教授和 Gaston Gnnot 教授设计研制。后来又被开发为商业软件,并且专门成立了 Maplesoft 公司负责 Maple 软件的开发和销售。与其他数学软件相比,Maple 的优势在于其强大的符号计算能力。Maplesoft 公司积极赞助符号计算的各种国际性学术会议,及时将新出现的高效算法地整合到 Maple 的函数包中。目前发布于 2007 年 2 月的版本 Maple 11(图 0-1),其功能主要有:

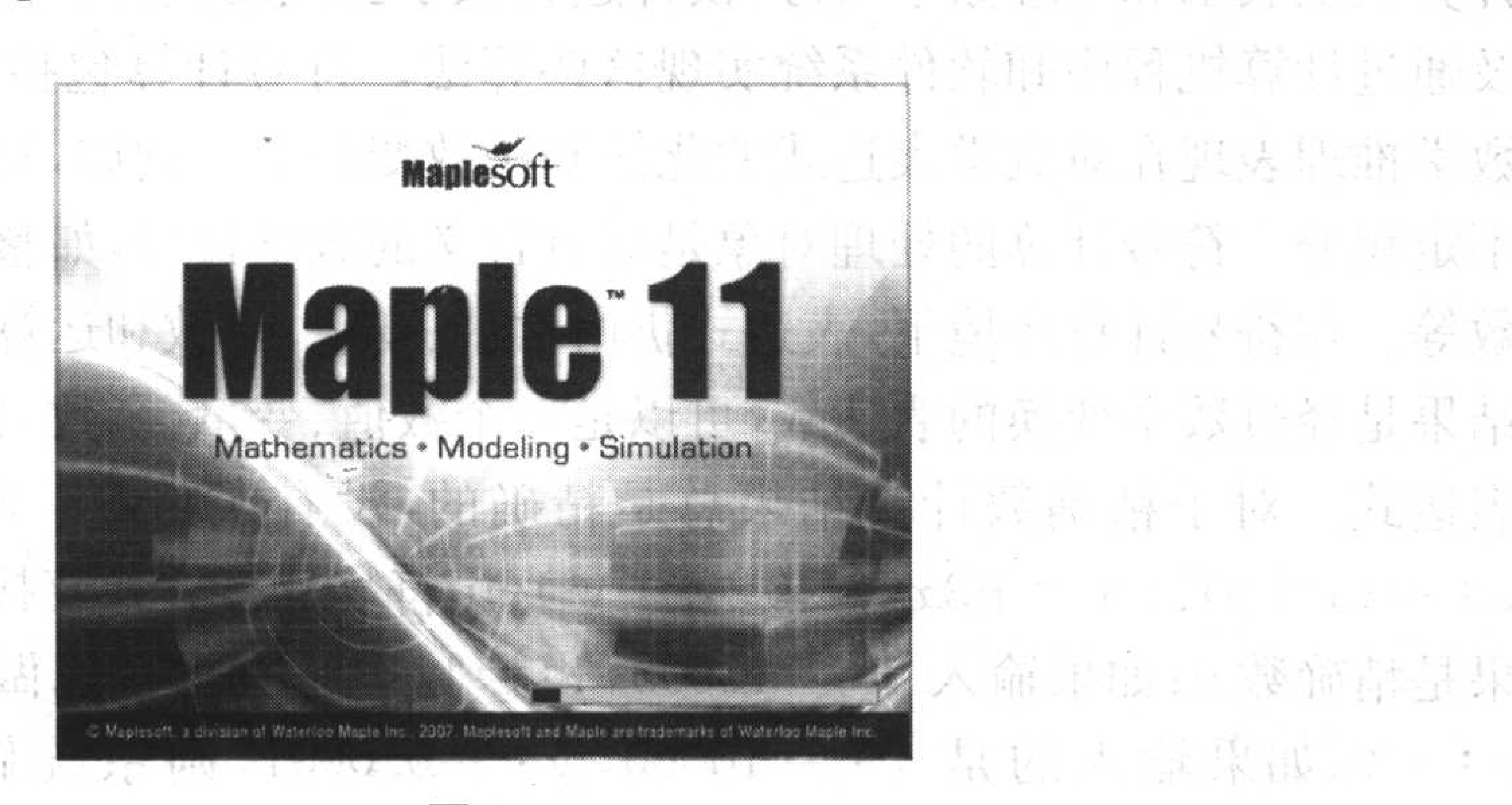


图 0-1

1. 解决数学问题

Maple 11 提供了约 300 多个任务模板和 4000 多个数学函数来解决常见的数学问题,涵盖了微积分、线性代数、微分方程、几何、群论、数论、图论、组合、运筹、优化、概率、统计等诸多领域。其中既包括无误差的符号计算,也包括高精度的数值计算,

以及二维、三维绘图。函数调用简单,用户可以像使用计算器一样地使用 Maple。

2. 生成科技文档

*.mw 格式的 Maple 文档可以将文本文字、数学公式、声音、图像等内容组合在一起,生成具有多媒体效果的专业科技文档。文档中还可以包含按钮、滑块、列表框等交互式组件,用户可以改变参数,重新计算结果。Maple 还提供诸如版面设计、拼写检查之类的编辑命令,用户可以像使用文字编辑软件一样地使用 Maple。

3. 应用程序开发

Maple 是一种面向对象的高级编程语言,有着严格的语法。针对不同硬件,Maple提供了经过优化的库函数。Maple 还是一个图形化的软件开发平台,用户可以方便地编写、调试程序,生成自己的函数包,从而将 Maple 的功能扩展。内容丰富的联机文档和帮助系统更是受到了用户的欢迎。

4. 辅助教学工具

针对大学教育这一市场,Maple 专门提供了 Student 函数包来帮助学生自学数学课程,加深对数学概念的认识。其内容包括微积分预修课、单变量微积分、多变量微积分、向量微积分、线性代数、数学辞典、绘图计算器、Maple T. A. 等。给学生提供了课后的教学助教,教师也可以从中发现比较好的课件。

0.3 初识 Maple

- 运行 Maple 11(maplew. exe),进入文件模式窗口(图 0-2)。

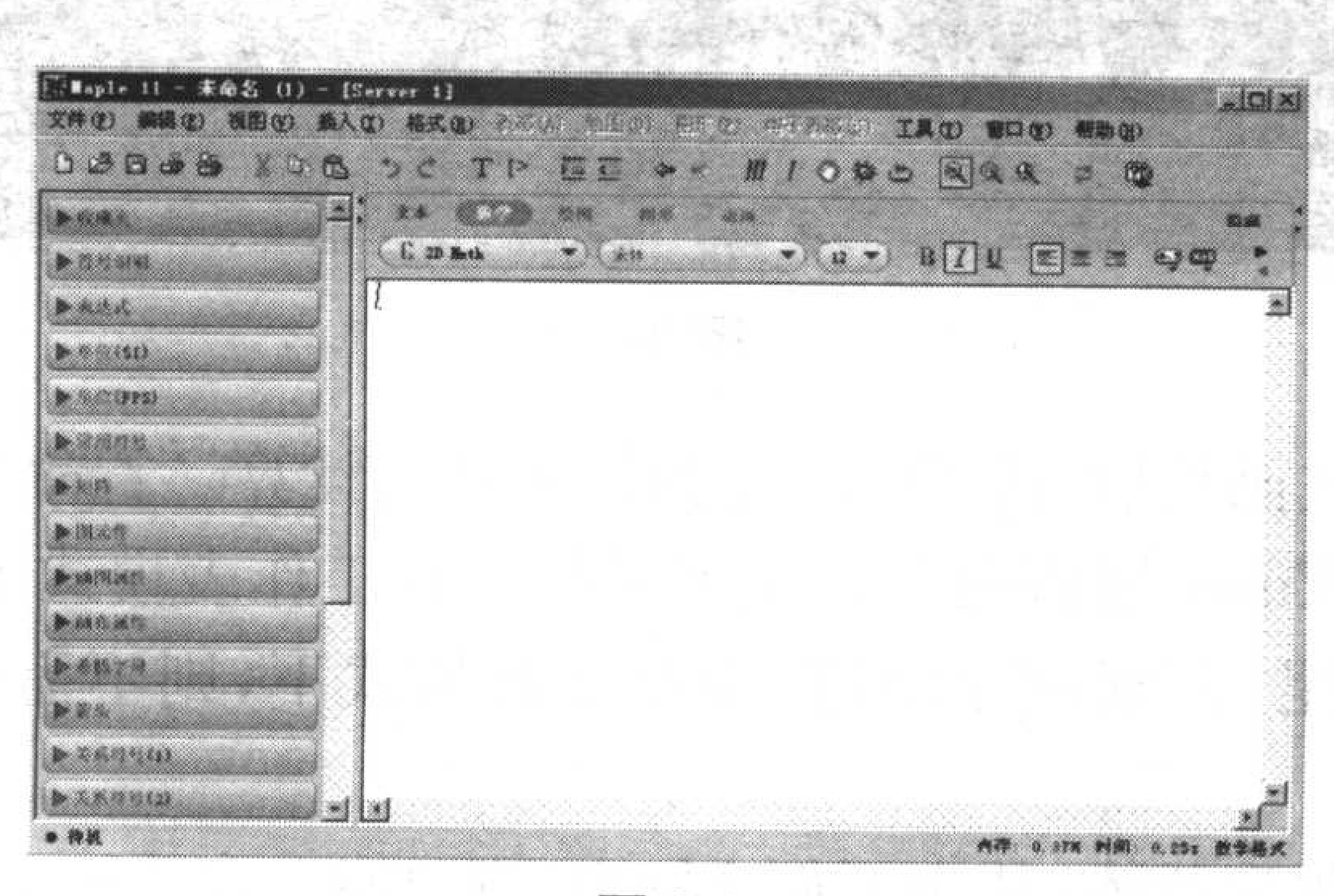


图 0-2

· 或者运行 Classic Worksheet Maple 11(cwmaple. exe),进入传统的工作表模式窗口(图 0-3)。

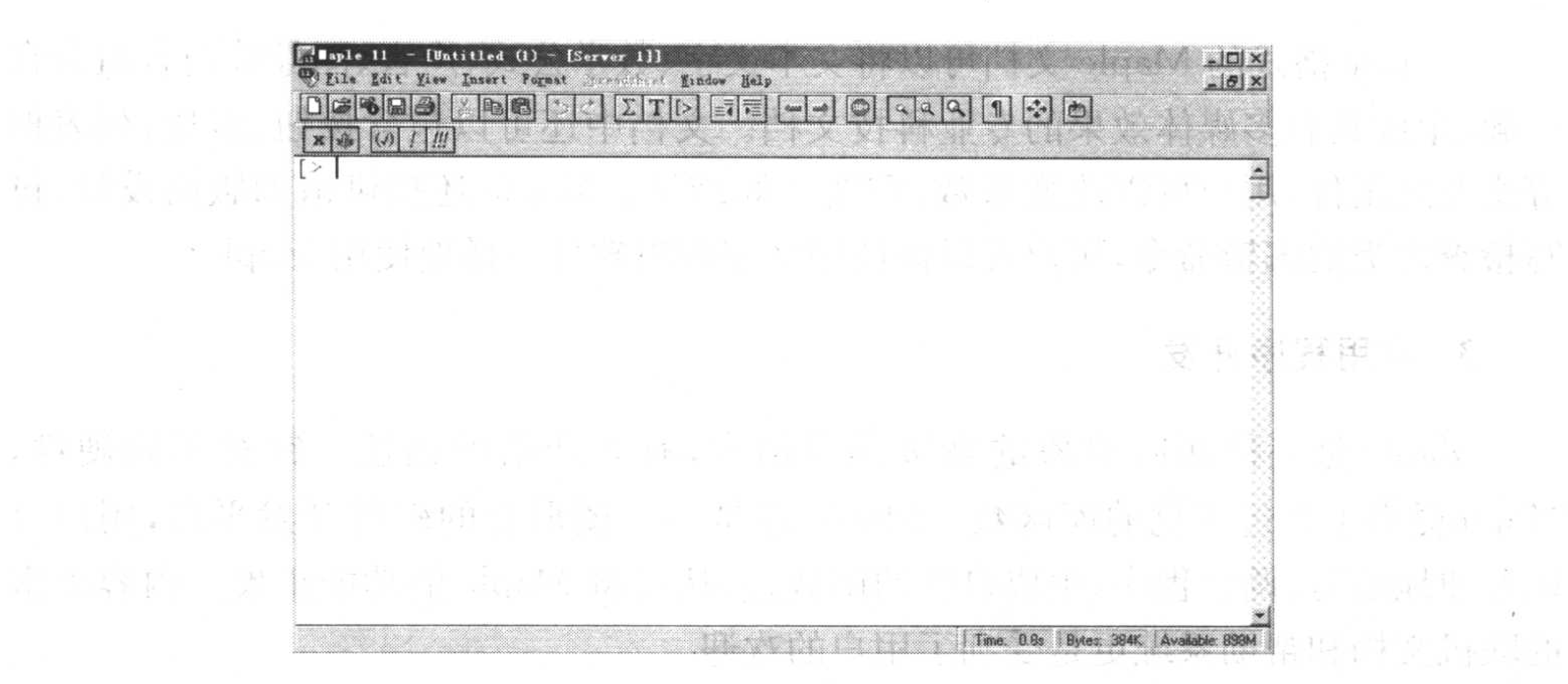


图 0-3

· 对于某些用户,也许他们更偏爱 Command-line Maple 11(cmaple. exe)那朴实无华的命令行模式窗口(图 0-4)。

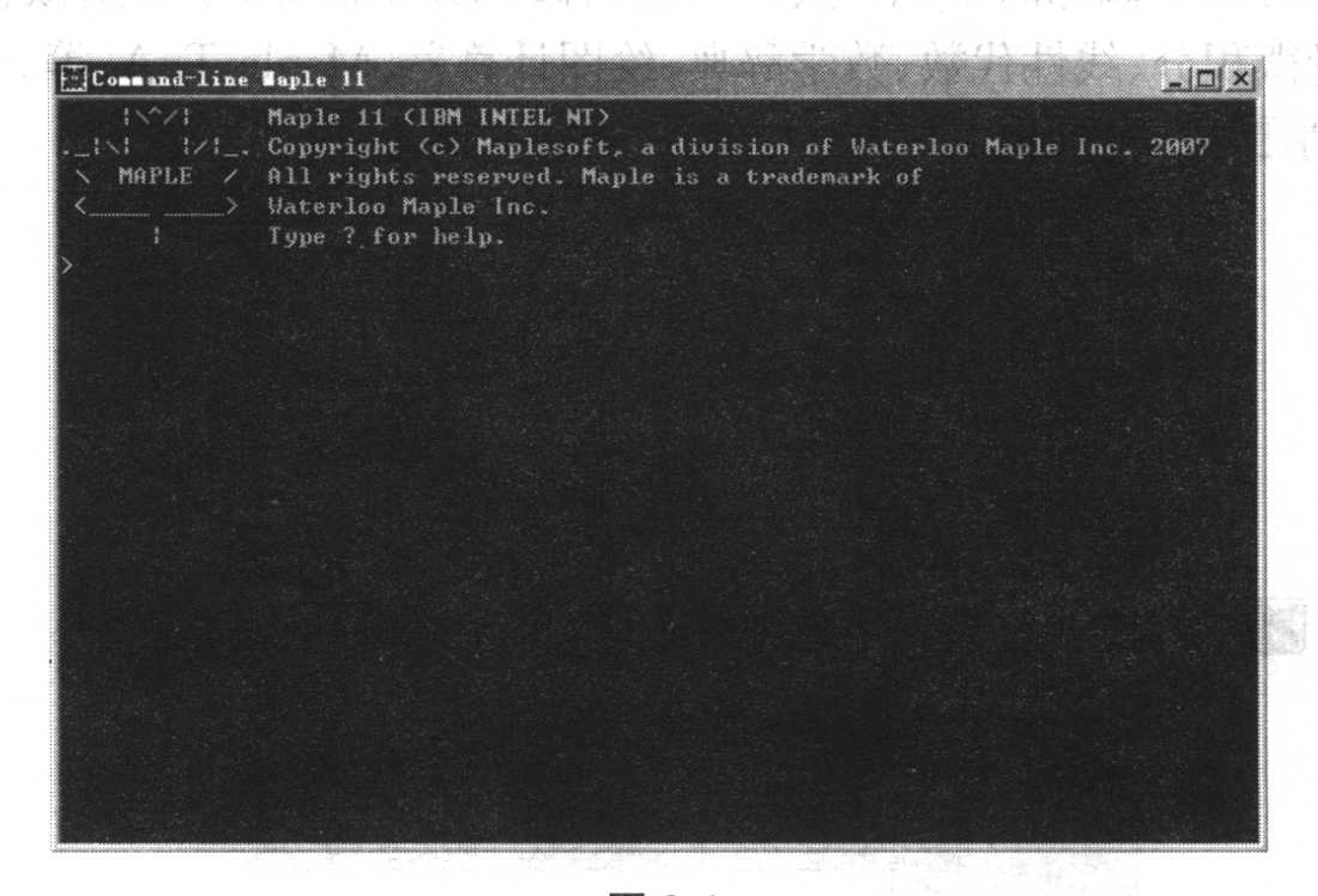


图 0-4

用户输入命令或函数后,按 Enter 键执行表达式计算。若命令很长,则可在输入过程中按下 Shift+Enter 键将命令分成若干行。在工作表模式下,表达式必须以冒号":"或分号";"结尾,否则程序出错;而在文件模式下,则不需要如此。如图 0-5,图 0-6 所示。

下面我们以例题为载体,从中了解 Maple 的功能。在 Maple 语句中,"#"及其

所在行后面的部分为注释说明语句。

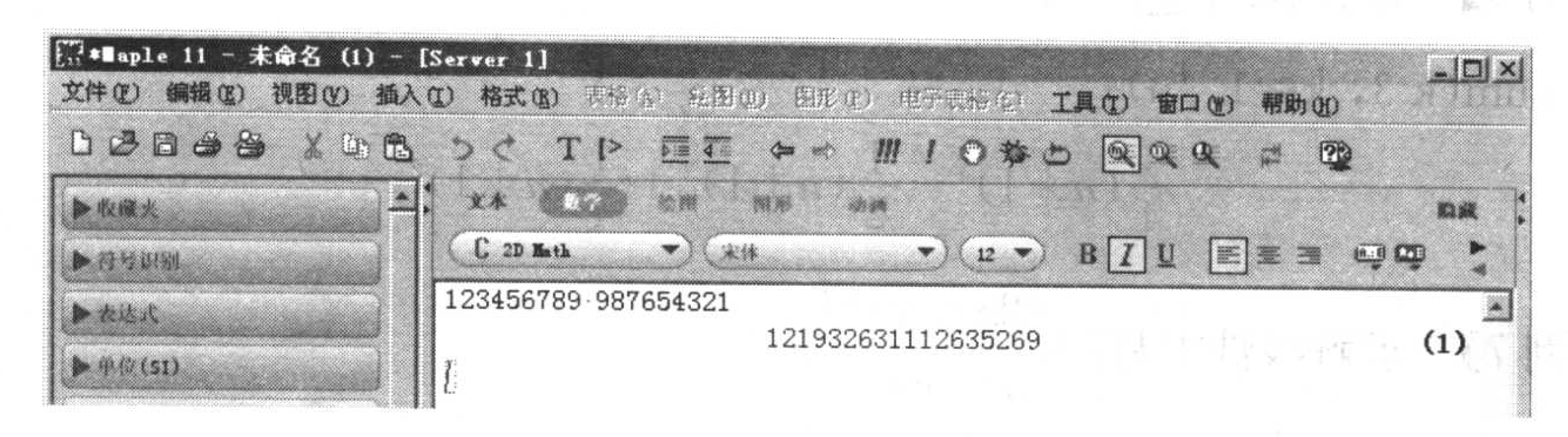


图 0-5

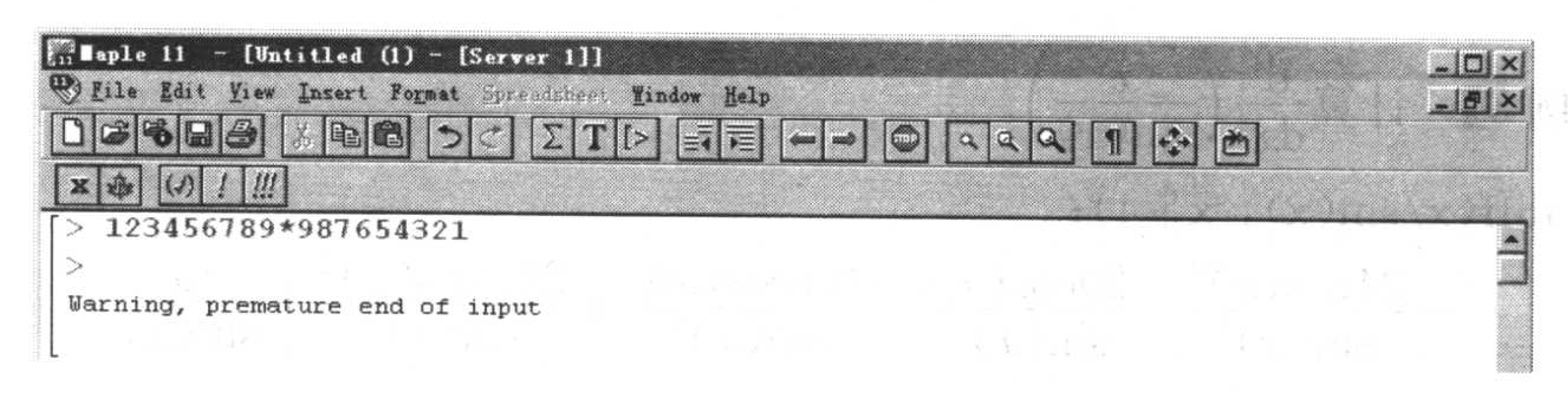


图 0-6

【例 1】 计算 9^{99}。

```
> 9^99;
```

29512665430652752148753480226197736314359272517043832886063884637676943433478020332709411004889

【例 2】 因式分解 $x^4+2x^3-13x^2-14x+24$。

```
> factor(x^4+2*x^3-13*x^2-14*x+24);
```

$$(x-1)(x+2)(x-3)(x+4)$$

【例 3】 展开多项式 $(x^2-1)(y-1)^3$。

```
> expand((x^2-1)*(y-1)^3); # 不要忘了输入两个括号之间的乘号
```

$$x^2y^3-3x^2y^2+3x^2y-x^2-y^3+3y^2-3y+1$$

【例 4】 计算 133 和 627 的最大公约数和 21,35,45 的最小公倍数。

```
> gcd(133,627);
```

19

```
> lcm(21,35,45);
```

315

【例 5】 计算和式 $\sum_{k=1}^{111} 2k-1$。

```
> add(2*k-1, k=1..111);
```

12321

【例 6】 计算和式 $\sum_{k=1}^{n} k^3$ 。

```
> sum(k^3, k=1..n);
```

$$\frac{1}{4}(n+1)^4-\frac{1}{2}(n+1)^3+\frac{1}{4}(n+1)^2$$

【例 7】 求解线性方程组$\begin{cases}2x+y=1\\x-y=5\end{cases}$。

```
> solve({2*x+y=1, x-y=5}, {x,y});
```

$$\{y=-3, x=2\}$$

【例 8】 计算$\frac{d^4}{dx^4}\left(\frac{x}{\sin(x)}\right)$。

```
> diff(x/sin(x), x$4);
```

$$-\frac{24\cos(x)^3}{\sin(x)^4}-\frac{20\cos(x)}{\sin(x)^2}+\frac{24x\cos(x)^4}{\sin(x)^5}+\frac{28x\cos(x)^2}{\sin(x)^3}+\frac{5x}{\sin(x)}$$

【例 9】 计算 $\int\frac{x}{x^4-1}dx$ 。

```
> int(x/(x^4-1), x);
```

$$\frac{1}{4}\ln(x-1)+\frac{1}{4}\ln(x+1)-\frac{1}{4}\ln(x^2+1)$$

【例 10】 计算 $\int_a^b\int_c^d(x^2+y^2)dxdy$ 。

```
> int(int(x^2+y^2,y=c..d),x=a..b);
```

$$\frac{1}{3}(d-c)(b^3-a^3)+\frac{1}{3}d^3(b-a)-\frac{1}{3}c^3(b-a)$$

【例 11】 计算矩阵 $A=\begin{pmatrix}1&2&3\\2&1&3\\1&2&3\end{pmatrix}$ 的特征值和特征向量。

```
> A:=Matrix([[1,2,3],[2,1,3],[1,2,3]]);
```

$$\begin{bmatrix}1&2&3\\2&1&3\\1&2&3\end{bmatrix}$$

```
> LinearAlgebra:-Eigenvectors(A);
```

$$\begin{bmatrix}0\\6\\-1\end{bmatrix},\begin{bmatrix}-1&1&1\\-1&1&\frac{-5}{2}\\1&1&1\end{bmatrix}$$

【例 12】 计算例 11 矩阵 A 的 1-范数 $\|A\|_1$。

> LinearAlgebra：－MatrixNorm(A,1)；

9

【例 13】 画出 $f(x)=x^2\sin(x)-1$ 在区间[－7,7]上的图像(图 0-7)。

> plot(x^2 * sin(x)－1，x=－7..7)；

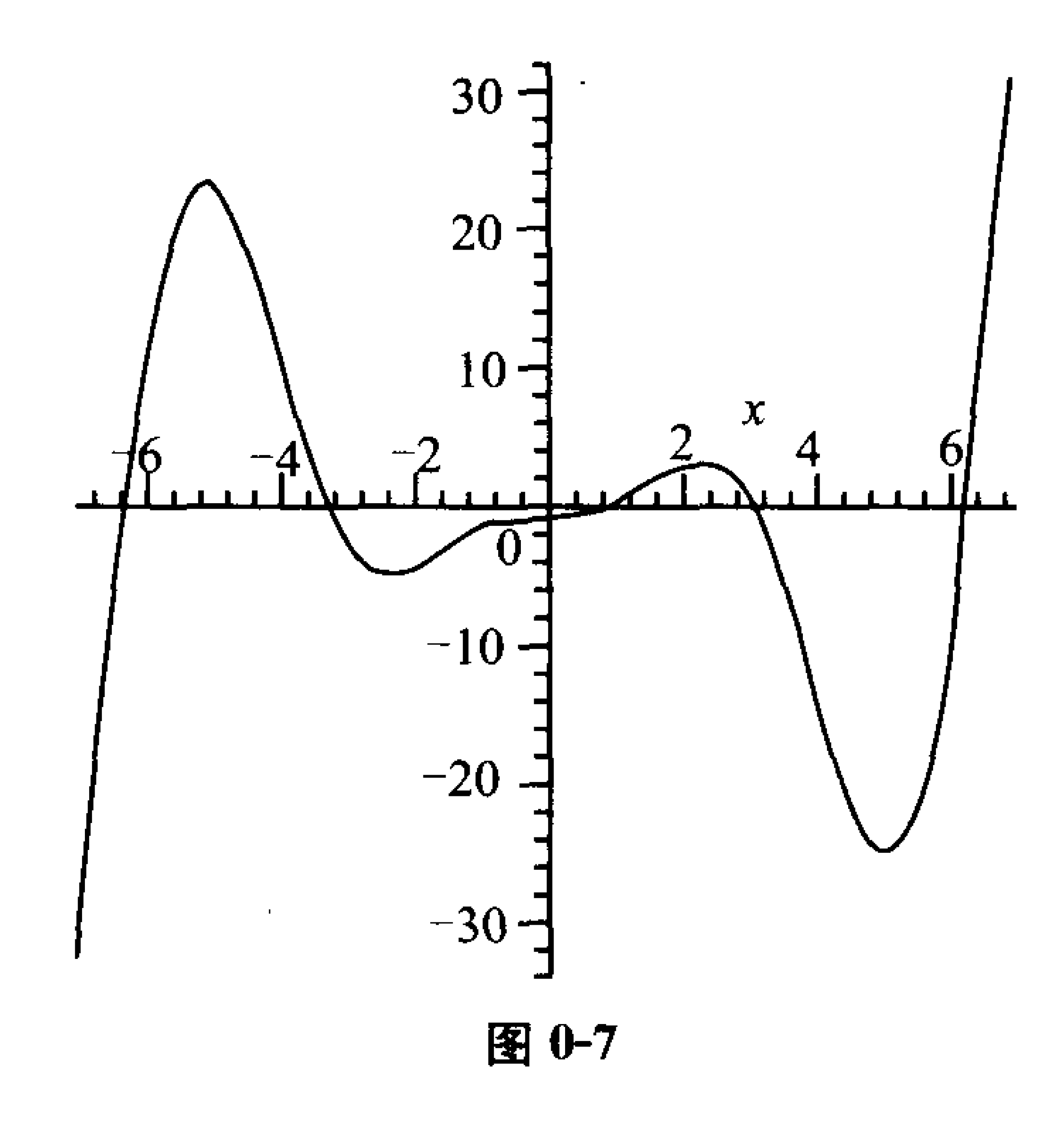


图 0-7

【例 14】 画长幅内次摆线：$\begin{cases}x=3\cos 2t+4\cos 3t\\ y=3\sin 2t-4\sin 3t\end{cases}$，$t\in[0,2\pi]$(图 0-8)。

> plot([3 * cos(2 * t)＋4 * cos(3 * t)，3 * sin(2 * t)－4 * sin(3 * t)，t=0..2 * Pi])；

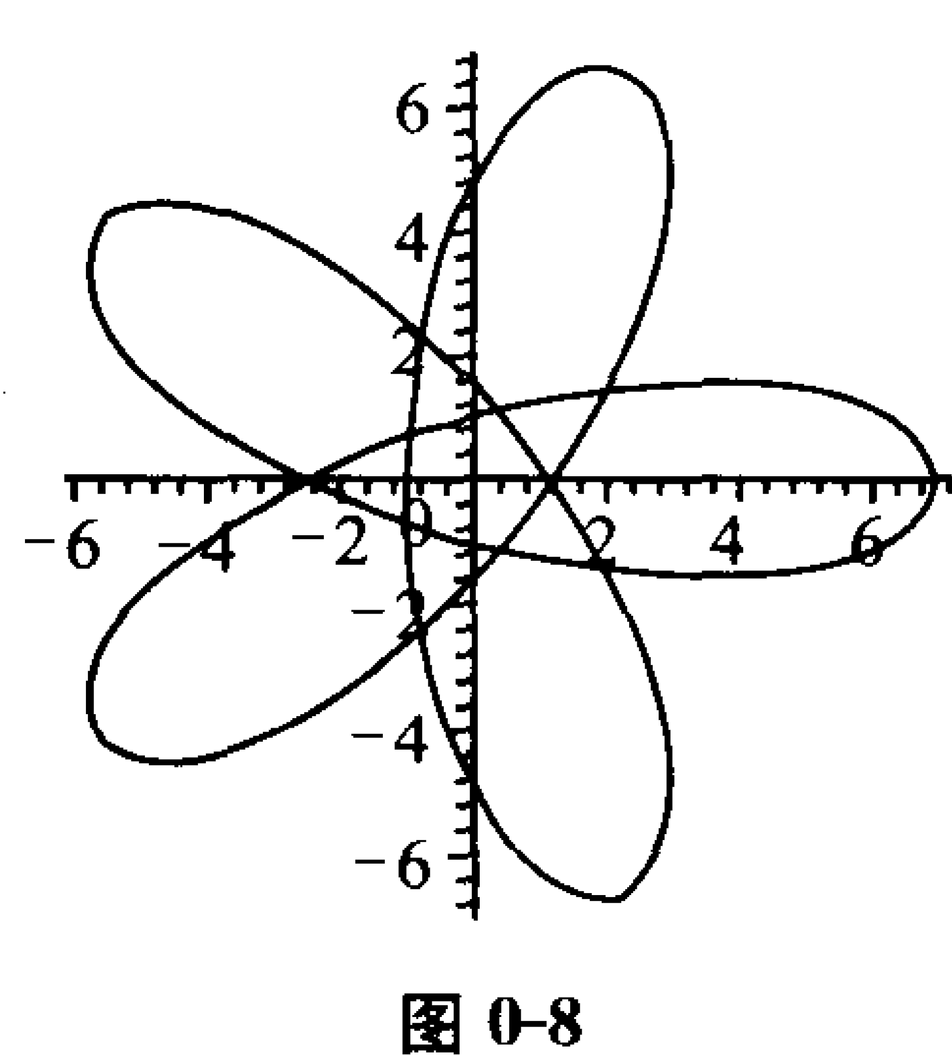


图 0-8

【例 15】 画出 $f(x,y)=xye^{-(x^2+y^2)}$ 在 $x\in[-2,2]$，$y\in[-2,2]$上的图像(图 0-9)。



```
> plot3d(x*y*exp(-x^2-y^2), x=-2..2, y=-2..2);
```

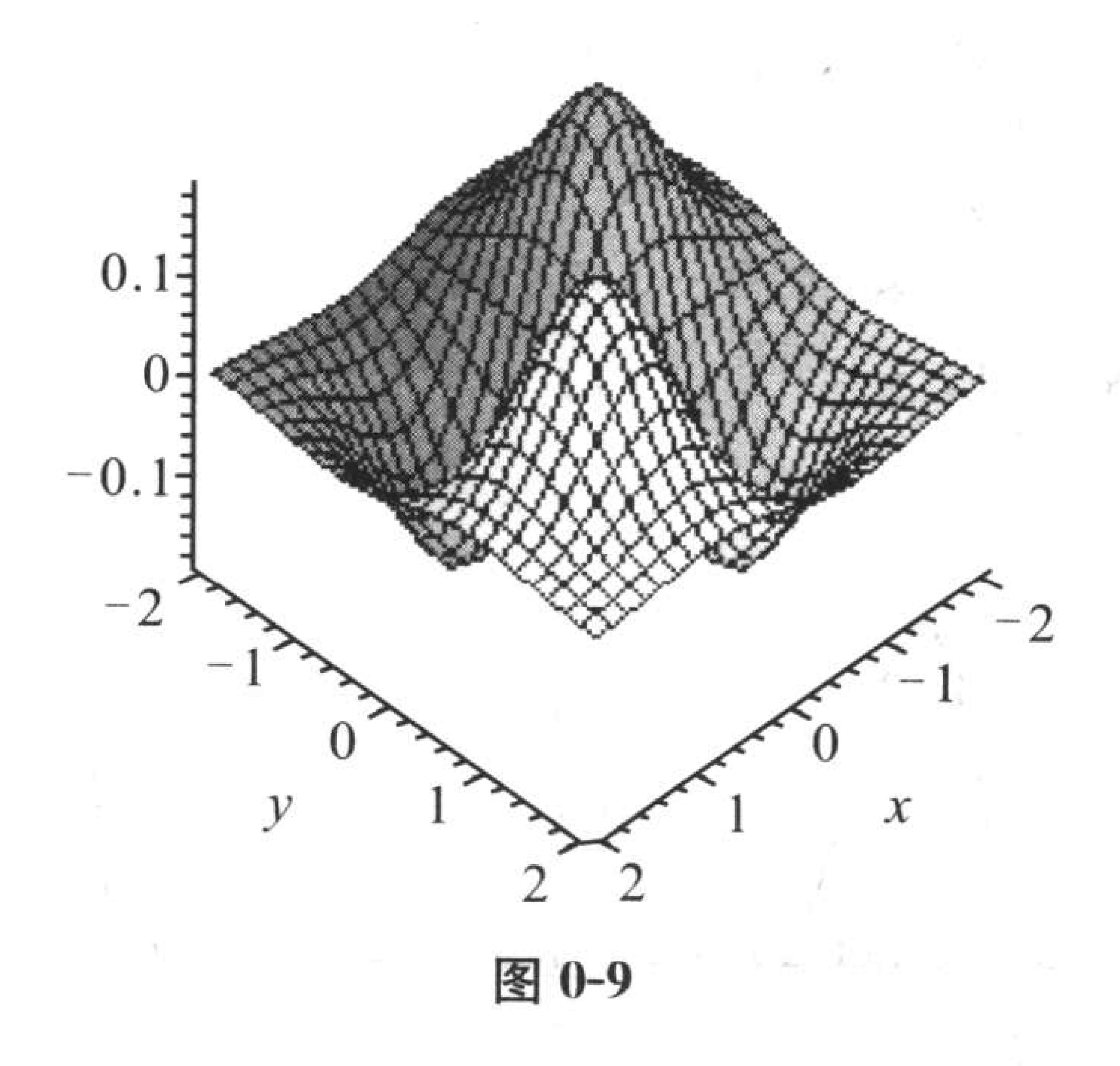


图 0-9

0.4 获取帮助

在 Maple 中，用户有多种方法获得帮助。本节中意在提供获取帮助的方法和途径，其效果取决于个人的尝试。部分例题放在第 8 章中。请注意，文件模式和工作表模式下的帮助形式并不完全相同。以下部分的帮助只在文件模式中有效。

1. 帮助菜单(Help Menu)

几乎所有的 Maple 联机帮助信息都可以在 Help 菜单中找到。对其中一些常用的菜单项，Maple 还提供了快捷键。例如：按 F1 键，将会弹出快速帮助窗口(图 0-10)。将光标停于需要帮助的词句处，然后按下 F2 键，将会显示出Maple关于此词句的所有帮助主题。按 Ctrl＋F1 键，将会弹出主帮助窗口(图 0-11)。

通过左边的目录表，用户可以找到相应的帮助主题。按 Ctrl＋F2 键，将会弹出快速参考窗口(图 0-12)。

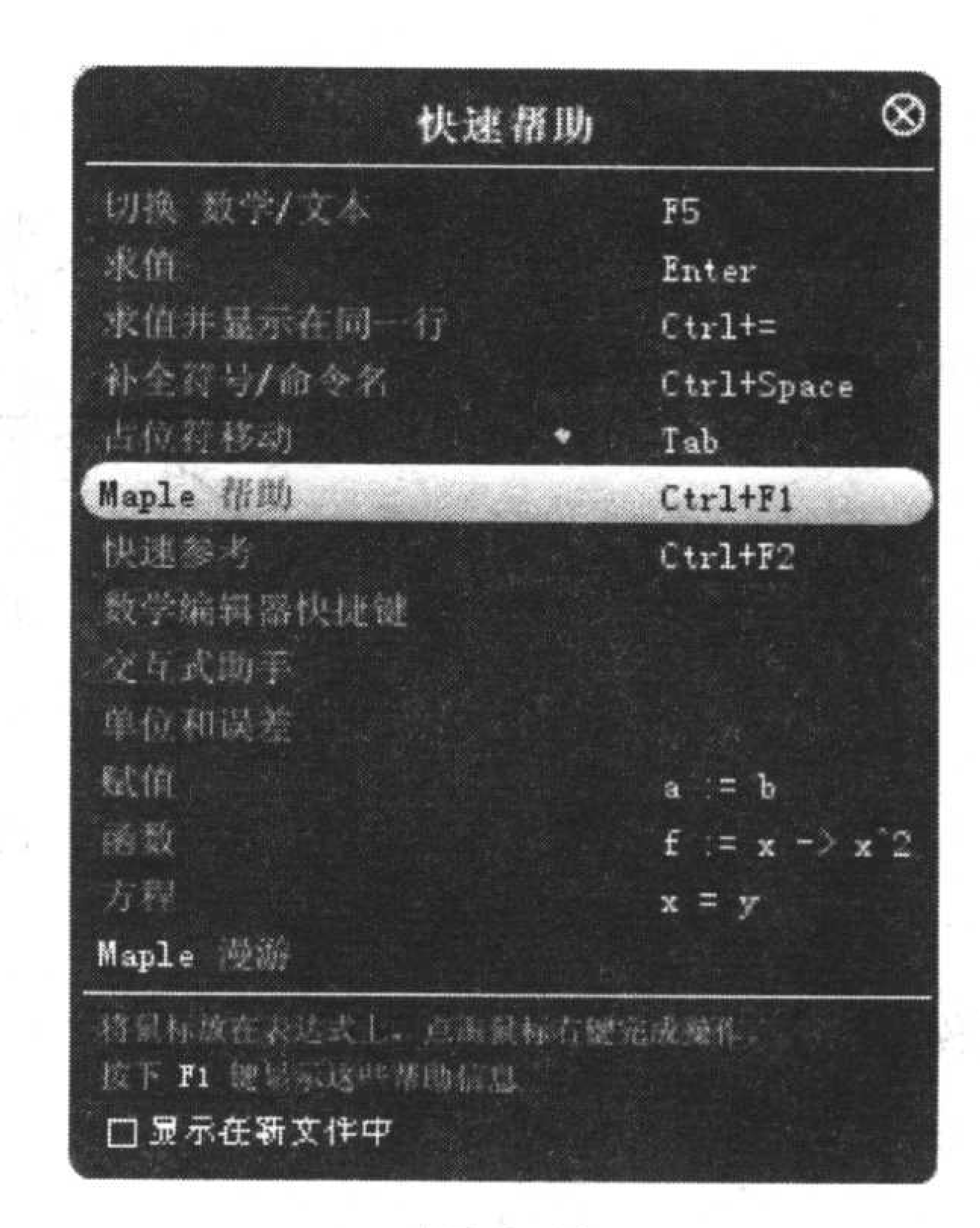


图 0-10

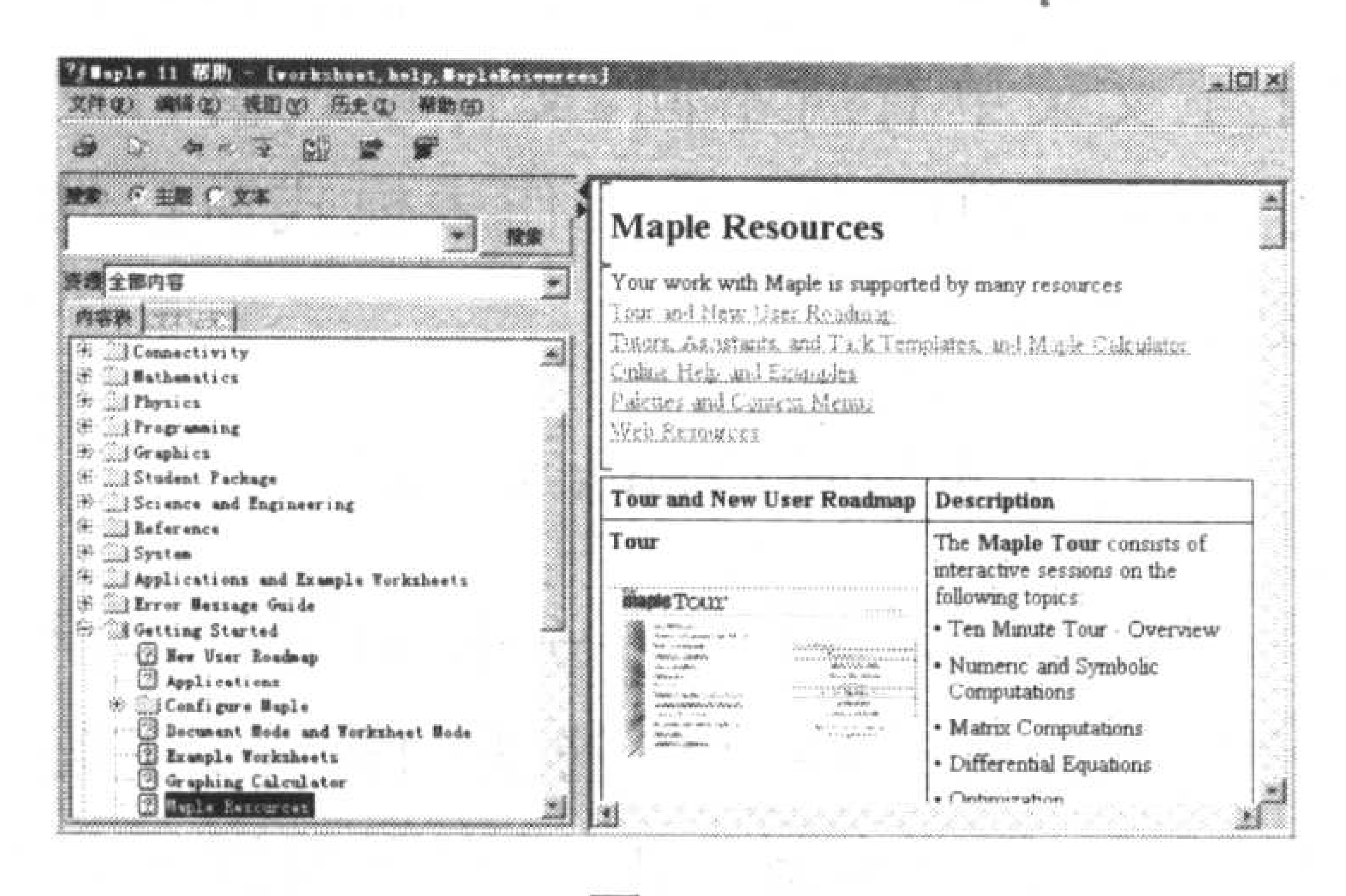


图 0-11

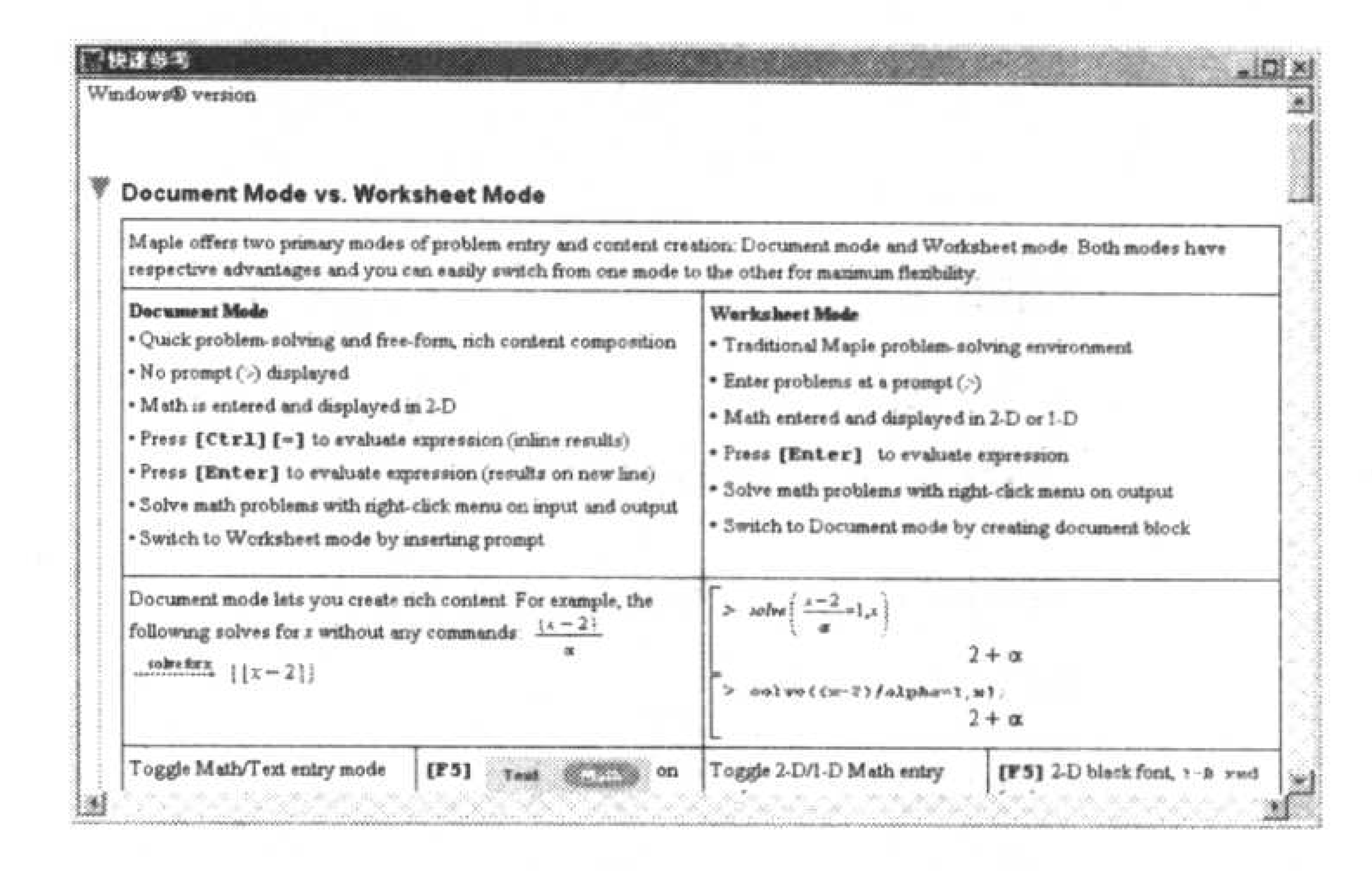


图 0-12

2. 快捷菜单(Context Menu)

在 Maple 工作区域的任何位置，按鼠标右键，都会弹出一个菜单。在不同位置，该菜单的内容也不完全相同，这种菜单称为快捷菜单或上下文菜单。

例如：我们输入

> plot3d(x^2－y^2，x=－1..1，y=－1..1)；

画抛物双曲面如图 0-13。用鼠标右键单击图像，弹出如图 0-14 的绘图菜单。而右击 plot3d 语句，则弹出如图 0-15 的菜单，此菜单包含有不少数学函数和常用的 Maple 命令，以方便用户输入。

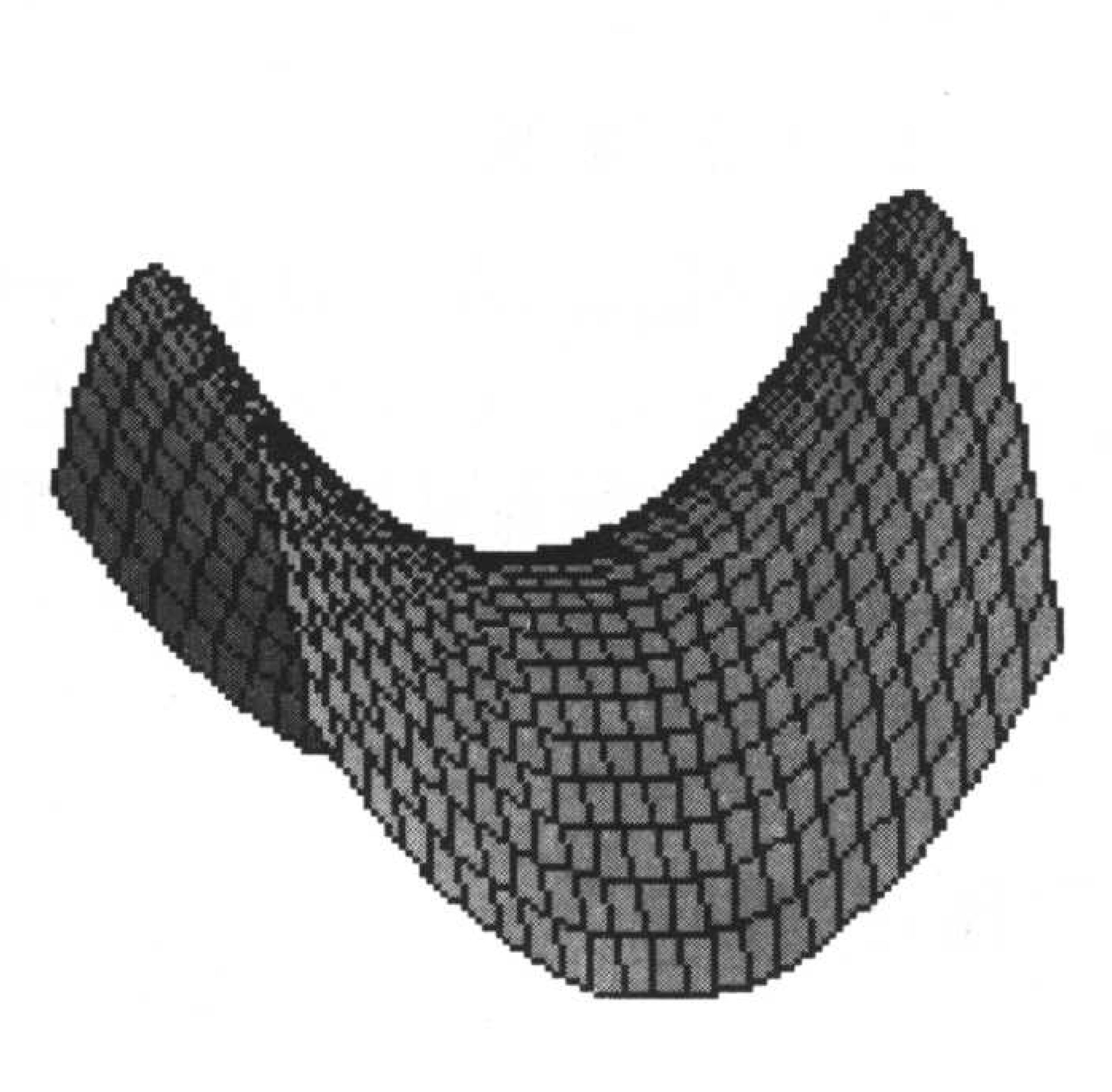


图 0-13

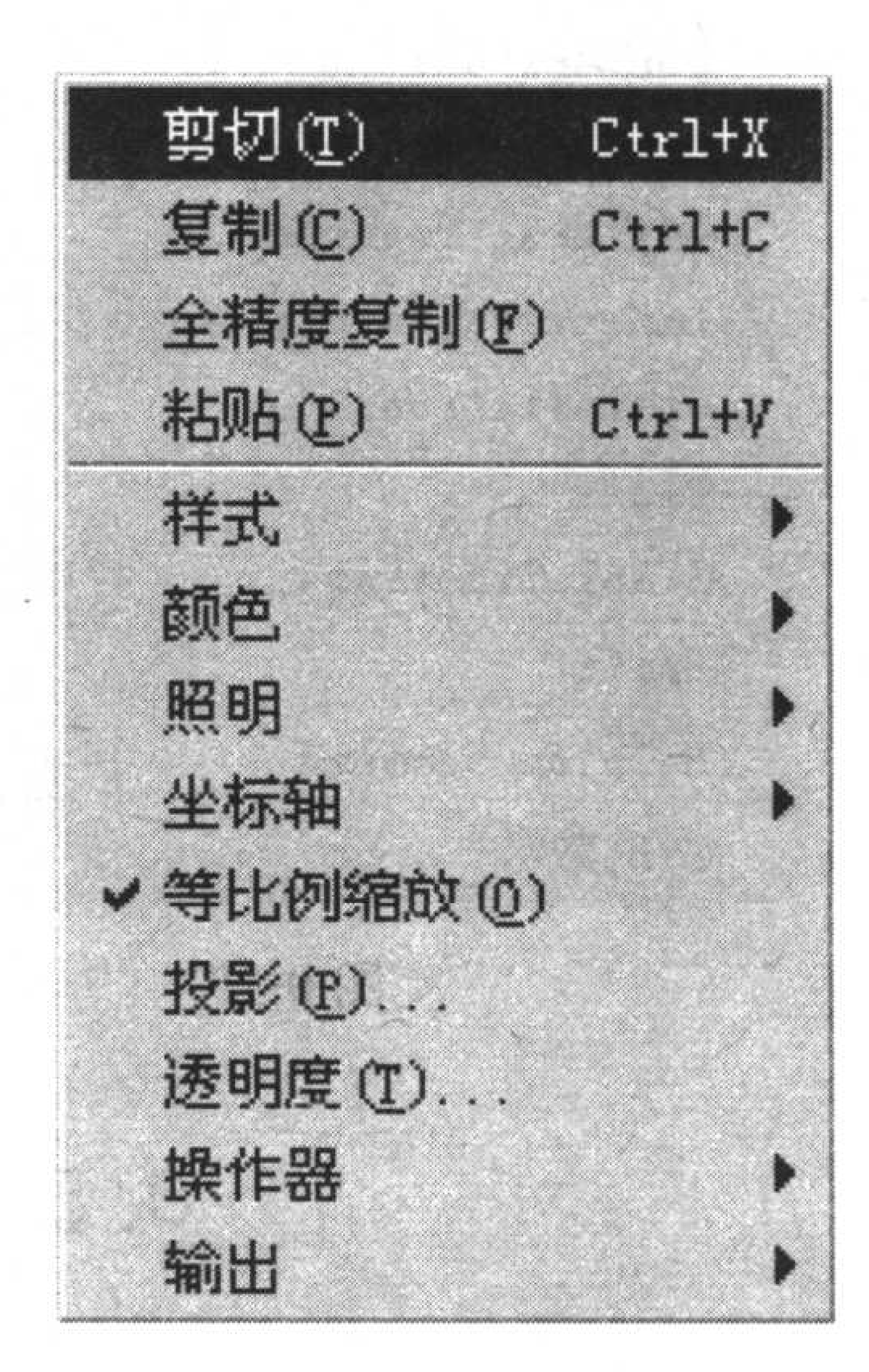


图 0-14

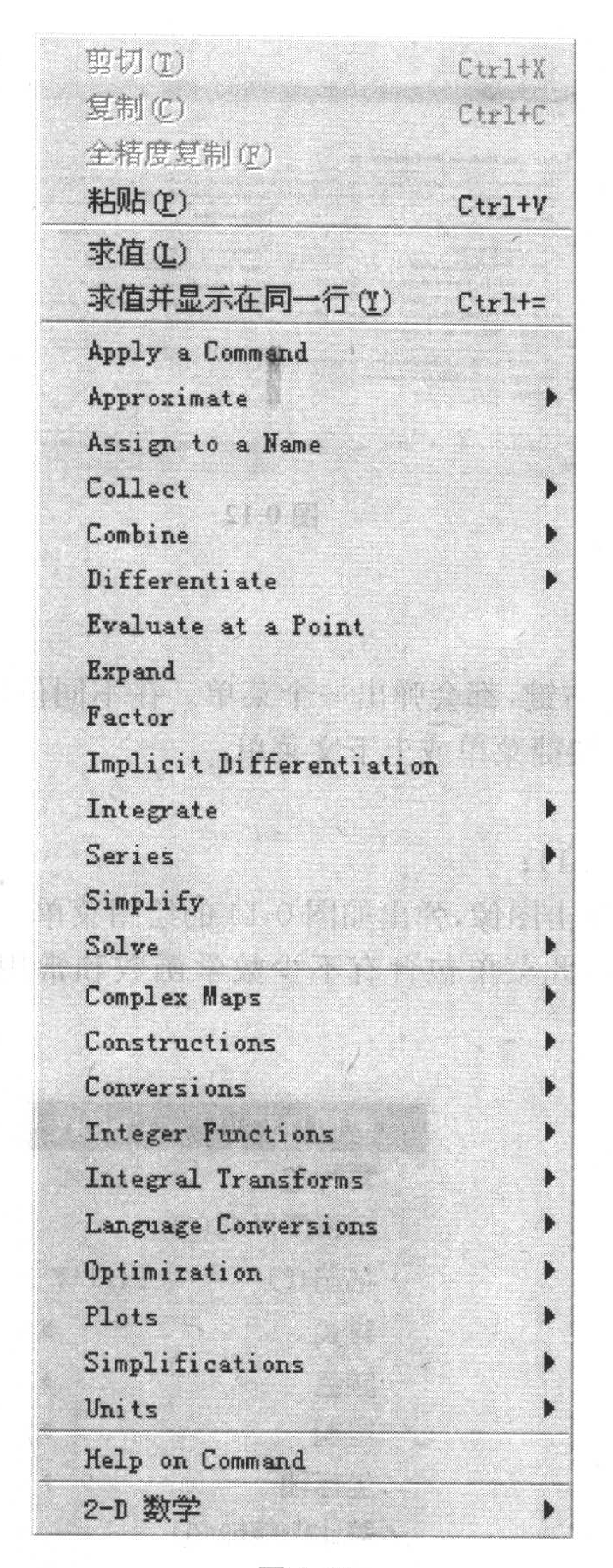


图 0-15

文件模式窗口和工作表模式窗口的上下文菜单有所不同,文件模式窗口的上下文菜单的内容比工作表模式窗口的更丰富。

3. 工具菜单(Tools Menu)

在菜单项"工具→分析助手"中,用户可以发现一些有用的辅助程序;在菜单项"工具→向导"中,用户则可以轻松地调用函数包 student 中的大量函数。

4. 在行文中获取帮助

对于偏爱 Command-line Maple 的用户来说,他们获得联机帮助的唯一途径是 ? 命令或 Help 命令。此命令也适用于文件模式和工作表模式。

例如:键入

```
> ? sin;
```

或

```
> help(sin);
```

则打开帮助窗口,显示三角函数的用法和例子。

5. 网络资源

访问 Maplesoft 公司的官方网站"http://www.maplesoft.com"来获取关于 Maple 的最新更新、用户文档和技术支持等。

0.5 库函数和函数包

Maple 11 提供了大量的过程和命令,统称为库函数。这些函数就位于主库(maple.mla)和 13 个函数包 Calculator.mla,CodeGeneration.mla,ContextMenu.

mla,Differential Geometry. mla,InstallerBuilder. mla, Maplets. mla, MmaTranslator. mla, Optimization. mla, Physics. mal, scandsea. mla, Statistics. mla, Student. mla,units. mla 中。主库包含绝大多数的常用数学函数,当 Maple 启动的时候,主库内的函数被自动加载,供用户直接调用。例如,计算积分 $\int \sin x \mathrm{d}x$,由于 int 函数属于主库函数,直接调用 int(sin(x),x)即可。

函数包中的函数,不会被自动加载,需要通过 with 命令或 use 语句来调用。在 Maple 中,有 3 种方式可以调用函数包中的函数。

1. with 命令

with 命令的常见用法有:

with(函数包)	调用函数包中的所有函数
With(函数包[子函数包])	调用子函数包中的所有函数
with(函数包,函数)	调用函数包中的特定函数
With(函数包[子函数包],函数)	调用子函数包中的特定函数

【例 16】 画圆心为(1,1)半径为 2 的红色实心圆(图 0-16)。

画圆函数 disk 在函数包 plottools 中,图像显示函数 display 在函数包 plots 中。

> with(plottools);

[arc, arrow, circle, cone, cuboid, curve, cutin, cutout, cylinder, disk, dodecahedron, ellipse, ellipticArc, hemisphere, hexahedron, homothety, hyperbola, icosahedron, line, octahedron, parallelepiped, pieslice, point, polygon, project, rectangle, reflect, rotate, scale, semitorus, sphere, stellate, tetrahedron, torus, transform, translate, vrml]

> with(plots);

[animate, animate3d, animatecurve, arrow, changecoords, complexplot, complexplot3d, conformal, conformal3d, contourplot, contourplot3d, coordplot, coordplot3d, densityplot, display, fieldplot, fieldplot3d, gradplot, gradplot3d, graphplot3d, implicitplot, implicitplot3d, inequal, interactive, interactiveparams, intersectplot, listcontplot, listcontplot3d, listdensityplot, listplot, listplot3d, loglogplot, logplot, matrixplot, multiple, odeplot, pareto, plotcompare, pointplot, pointplot3d, polarplot, polygonplot, polygonplot3d, polyhedra_supported, polyhedraplot, rootlocus, semilogplot, setoptions, setoptions3d, spacecurve, sparsematrixplot, surfdata, textplot, textplot3d, tubeplot]

```
> display(disk([1,1],2,color=red));
```

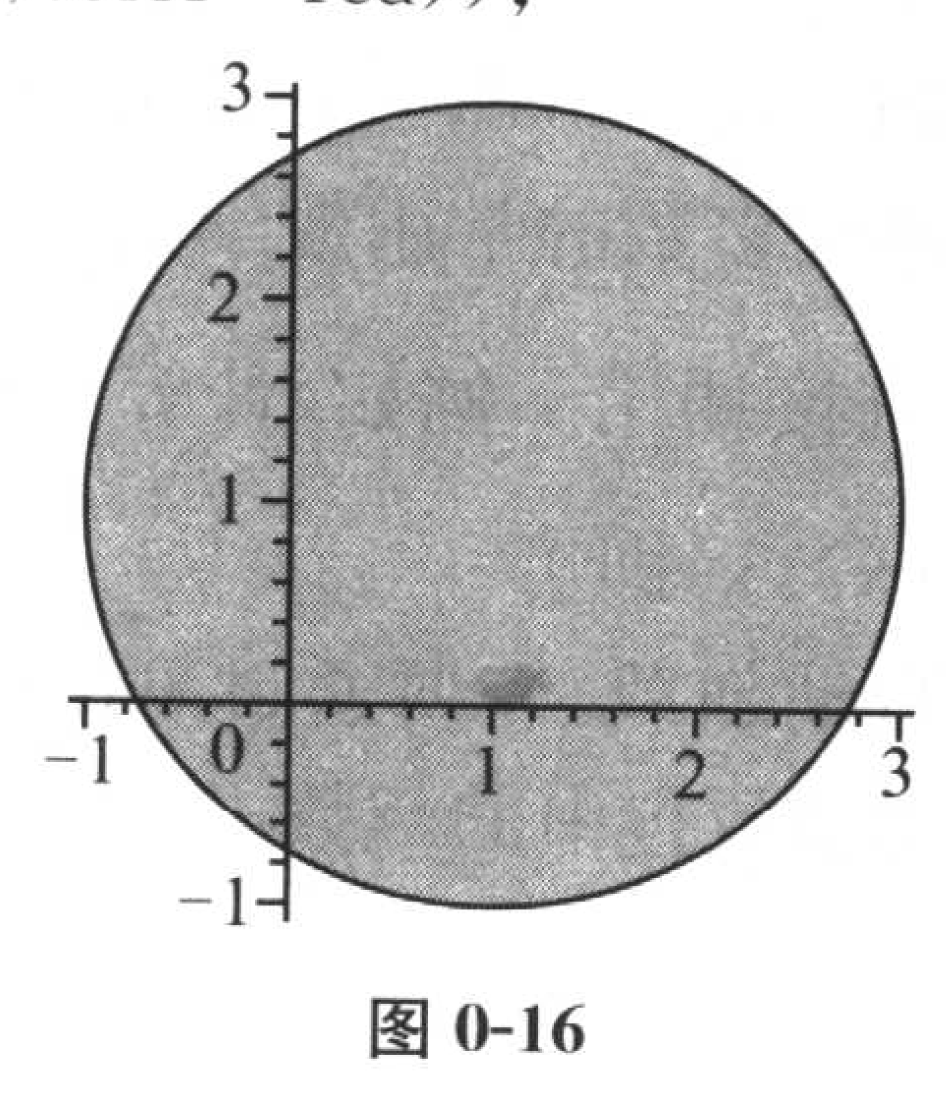


图 0-16

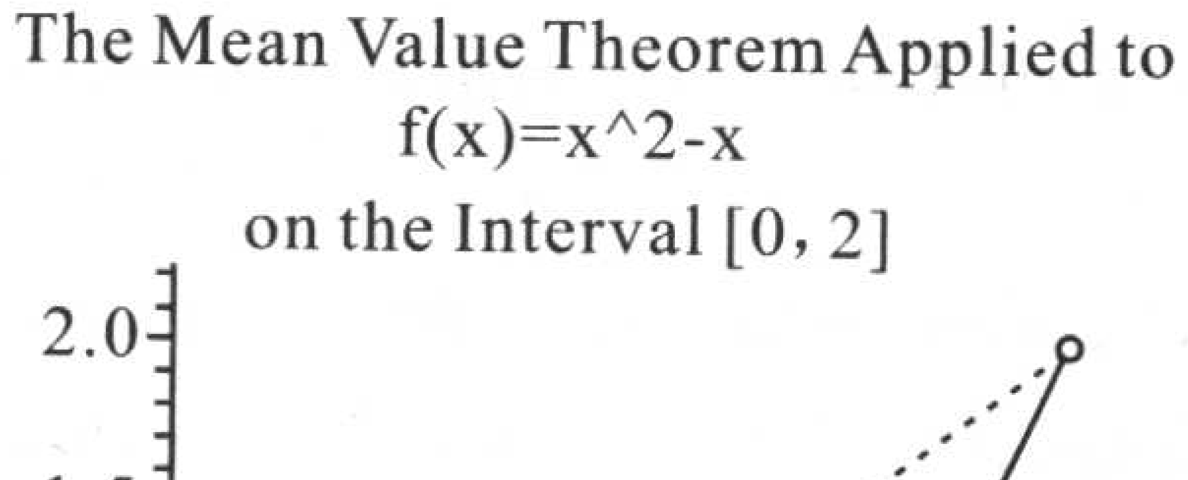
2. use 语句

use 语句的一个常见用法为：

模块:-函数名

【例 17】 演示单变量微积分的中值定理(图 0-17)。

```
> Student:-Calculus1:-MeanValueTheorem(x^2-x, x=0..2,output=plot);
```



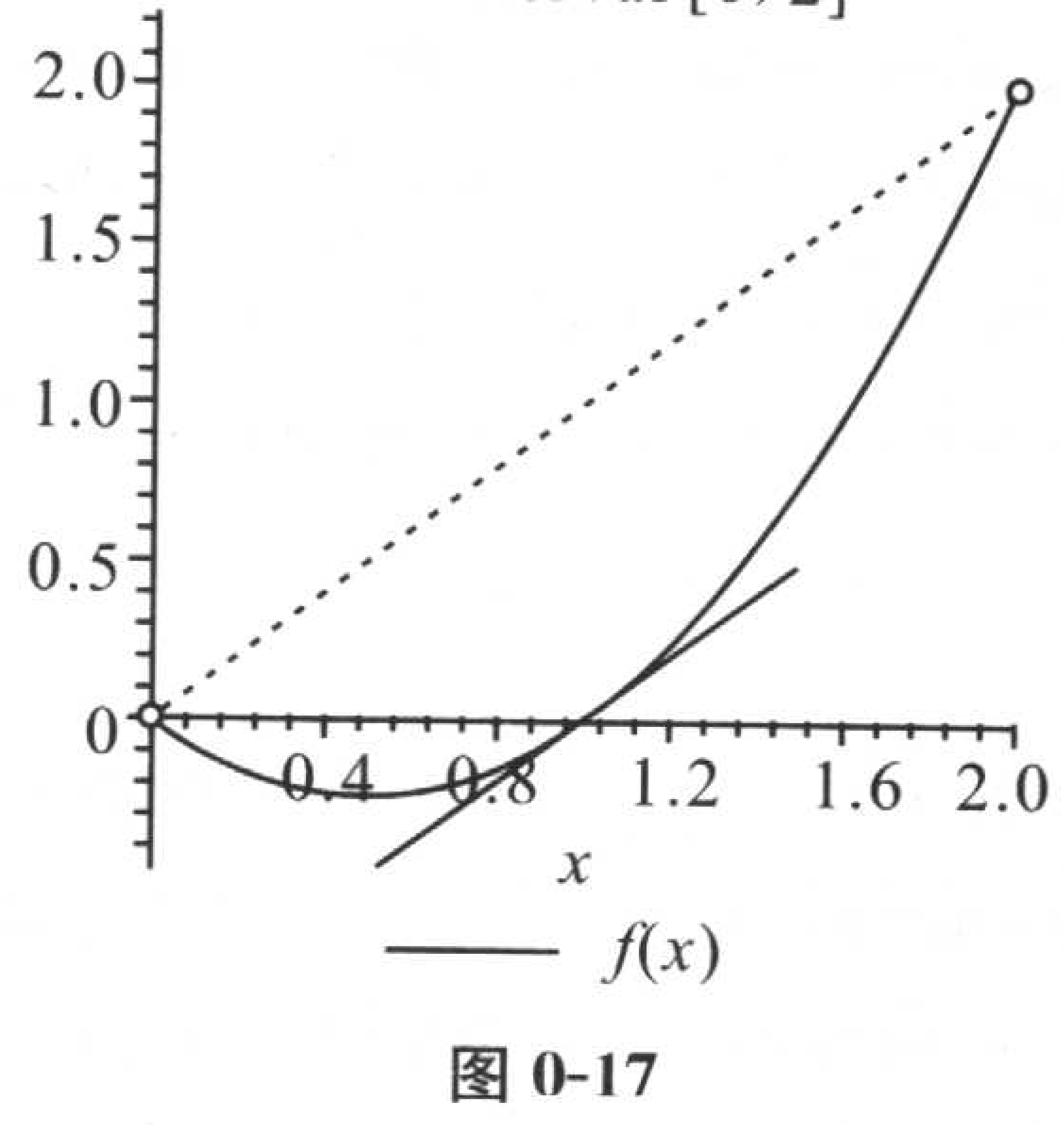


图 0-17

3. 直接引用函数包

函数包[函数](参数)
函数包[子函数包][函数](参数)

【例 18】 产生二阶随机整数矩阵。

```
> LinearAlgebra[RandomMatrix](2,2);
```

$$\begin{bmatrix} 44 & -31 \\ 92 & 67 \end{bmatrix}$$

此语句与 LinearAlgebra：－RandomMatrix(2,2)有着相同的效果。

表 0-1 列出了部分常用的函数包以供参考。

表 0-1　部分常用函数包

函数包	说明
ArrayTools	数组处理
AudioTools	音频处理
combinat	组合数学
CurveFitting	曲线拟合
DEtools	常微分方程
DiscreteTransforms	离散 Fourier 变换
Domain	整环
FileTools	文件处理
finance	金融
geom3d	立体几何
geometry	平面几何
Groebner	计算 Gröbner 基
group	群论
ImageTools	图像处理
IntegrationTools	符号积分
inttrans	积分变换
LibraryTools	函数库处理
LinearAlgebra	线性代数
ListTools	列表处理
Logic	逻辑
LREtools	线性递归方程求解
MatrixPolynomialAlgebra	多项式矩阵运算

续表

函数包	说明
networks	图网络
numapprox	数值逼近
numtheory	数论
Optimization	优化
OrthogonalSeries	正交多项式
PDEtools	偏微分方程
plots,plottools	绘图
PolynomialTools	多项式
powseries	幂级数
RandomTools	生成随机数
RootFinding	方程求数值根
simplex	单纯形
SNAP	混合计算
SolveTools	方程求代数解
Statistics	统计
StringTools	字符串处理
Student	大学数学教学包
Student[Calculus1]	大学数学—单变量微积分
Student[LinearAlgebra]	大学数学—线性代数
Student[MultivariateCalculus]	大学数学—多变量微积分
Student[Precalculus]	大学数学—微积分预修课
Student[VectorCalculus]	大学数学—向量微积分
sumtools,SumTools	级数求和
VariationalCalculus	变分法
VectorCalculus	向量微积分

第 1 章　Maple 的基本量

1.1　数与数的表示

1.1.1　简单数值类型

Maple 的简单数值类型有整数、有理数、实数和复数(表 1-1)。

表 1-1　简单数值类型

类　型	描　述	示　例	说　　明
integer	整 数	1234567	任意长度整数
faction	有理数	13243/5768971	既约分数
float	实 数	1234567.0	任意精度小数
complex	复 数	1234567.0+2.3I	实部和虚部可为整数、有理数、实数
string	字符串	"Hello, World!"	在 32 位操作系统中,至多包含 $2^{28}-17$ 个字符; 在 64 位操作系统中,至多包含 $2^{35}-33$ 个字符

整数(integer)由若干个 0 到 9 的数字组成,数字之间不能有空格、逗号和其他字符,正负号放在整数的首位,输入时正号可省略不写。只要内存允许,原则上 Maple 可以表示任意长度的精确整数,不受所用计算机的字长限制,实际上受控于计算机的内存容量。整数与整数的运算结果仍是准确的整数或有理数。例如:3 的 123 次方为 59 位十进制整数。

整数还可细分为:负整数(negint)、正整数(posint)、非负整数(nonnegint)、非正整数(nonposint)、偶数(even)和奇数(odd)。

【例 1】 整数运算。

```
> 3^123;
```

48519278097689642681155855396759336072749841943521979872827

```
> length(%); # 给出上一次输出的位数,%表示上一次输出
```

59

```
> kernelopts(maxdigits); # 查看本系统整数的最大位数
```

268435448

```
> is(12347, prime); # 12347 是否为素数
```

true

```
> convert(12,binary); # 转化为二进制数
```

1100

```
> convert(17,octal),convert(17,hex); # 转化为八进制、十六进制数
```

21,11

有理数(fraction)也是符号计算系统的数据成员。有理数是准确数。当两个整数相除而又不能整除时,系统就用有理数来表示。例如:$\frac{2}{3}$。输入时,我们用分子/分母的形式,用除号隔开分子与分母,在文件模式窗口下,如果输入"x/"则显示为分子在上分母在下的形式"$\frac{x}{}$";输出时,系统对分数做可能的自动化简,约去分子和分母的公因子,再用化简过的分式形式输出。有理数之间的运算结果仍然是准确的有理数或整数。

【例 2】 分数运算。

```
> 123/456+567/789;
```

$$\frac{39511}{39976}$$

用函数 evalf 或 value 将整数或有理数转换为实数。

```
> evalf(12+2/3+a);
```

$$12.66666667+a$$

实数(float)是用浮点数表示的,Maple 中的实数可取任意位的有效位数,默认值取小数点后 10 位有效数字。这是一种具有任意精确度的近似实数。计算中也可以保持和控制实数的精度。输入时实数有两种表示方法:一种是小数形式,例如:.679,0.,−1234.6 都是合法的小数;另一种是指数形式即科学记数法,例如:0.1234e5 表示 123450.。

实数可以和整数、有理数混合计算,其结果仍是一个实数。

【例 3】 实数运算。

```
> 1+1/2+0.3;
                    1.800000000
> 3e3,1.3E3,-.3e-3;
              3000., 1300., -0.0003
> convert(0.123456789,fraction); # 小数转换为分数
```

$$\frac{1356659}{10988938}$$

```
> float(1.3,-5);
                    0.000013
```

复数(complex)由实部和虚部组成。实部和虚部可用整数、有理数和实数表示。在 Maple 中,大写字母 I 表示虚数单位 $\sqrt{-1}$。

【例 4】 复数运算。

```
> sqrt(-9),exp(3.+7*I);
              3I, 15.14253157+13.19592859I
> x:= sin(1.1+2.2*I);
              x:=4.070953523+2.021725618I
> Re(x),Im(x); # 取 x 的实部和虚部
              4.070953523, 2.021725618
```

字符串(string)是用双引号引起来的字符序列,函数包 StringTools 中约有 180 个有关字符串操作的命令供用户使用。

【例 5】 字符串运算。

```
> ta:="abc": tb:="xyz": cat(ta,tb);
                    "abcxyz"
```

1.1.2 数学常数

在 Maple 中已经定义了一些数学常数(表 1-2)。例如:圆周率 π,无穷大∞,欧拉常数 γ 等。命令 constants 可显示出这些常数。数学常数可以直接用在公式推导中,Maple 中的数学常数都是准确数;用于数值计算的数学常数可以取任意精度。函数包ScientificConstants 中则定义了更多的物理、化学常数。系统也允许用户添加自定义的常数。

表 1-2　数学常数

常　　数	说　　明
Catalan	Catalan 常数 $\sum_{k=0}^{\infty}\frac{(-1)^k}{(2k+1)^2}\approx 0.915965594\cdots$
false, true	逻辑值假、真
FAIL	不能确定真假的逻辑值
gamma 或 γ	欧拉常数 $\lim_{n\to\infty}\sum_{k=1}^{n}\frac{1}{k}-\ln(n)\approx 0.5772156649\cdots$
I	虚数单位 $\sqrt{-1}$
Infinity 或∞	无穷大
Pi、pi 或 π	圆周率(pi 用于符号计算,Pi 用于数值计算)
undefined	不确定值(Maple 的保留字之一)

【例 6】　有关的数学常数。

```
> constants;
```

$$\text{false},\ \gamma,\infty,\ \text{true},\ \text{Catalan},\ \text{FAIL},\ \pi$$

```
> [Pi+1/3,evalf(Pi+1/3)]; # 输入 Pi 运行后显示为 π
```

$$\left[\pi+\frac{1}{3},\ 3.474925987\right]$$

```
> evalf(pi+1/3); # evalf 对 pi 无效
```

$$\pi+0.3333333333$$

1.2　变　　量

1.2.1　给变量取名

Maple 的变量名由标识符表示,标识符以字母开头,后跟字母、数字或下划线,变量名的字符长度不限。例如:AbCdEfGhIjK,Win32,ΔABC,X_Y_Z 都是合法的变量名;而 2t 和 u v(u 与 v 之间有一空格)不能作为变量名。变量名中英文字母大小写意义不同,因此 A 与 a 表示两个不同的变量。例如,用户可用大写字母 A 表示矩阵变量,用小写字母 a 表示矩阵元素变量,通常按数学表示习惯给变量取名。

如果要用希腊字母作变量名,输入希腊字母的英文名称,或展开“希腊字母”面

板，选择所需字母(图 1-1)。例如：

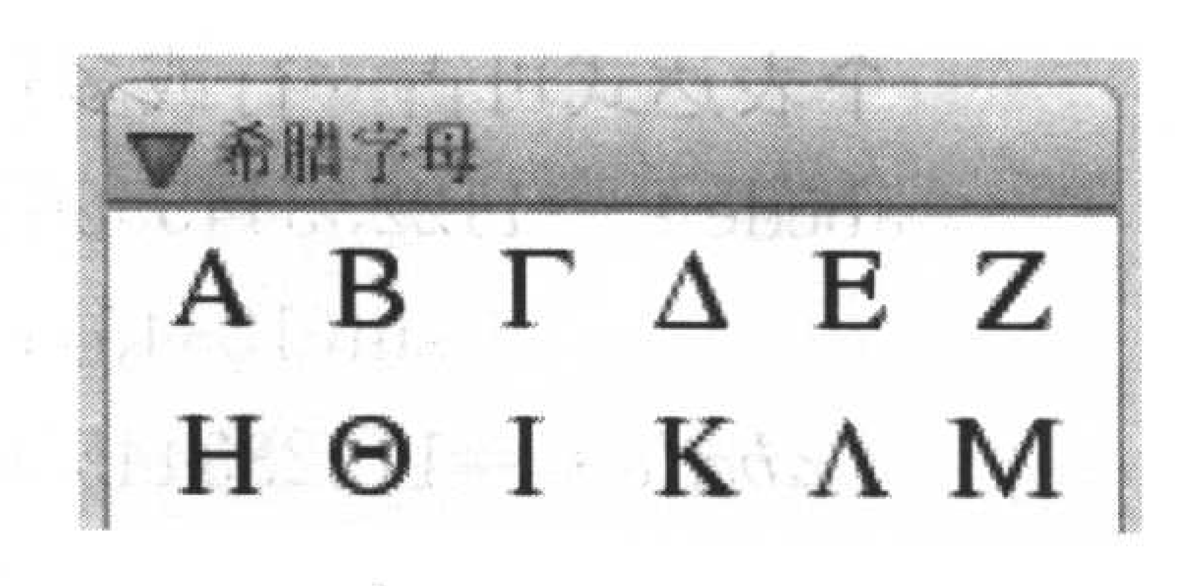


图 1-1

```
> alpha := pi+12;
```

$$\alpha := \pi + 12$$

请不要用 Maple 中的函数名和保留字作变量名，如 sin，and，if，fi，od 等。如果你误用了 Maple 的保留字，系统会给出相应的提示。

数值有类型，变量也有类型。在 Maple 中变量随取随用，通常在运算中不需要对变量进行类型说明，系统根据你对变量所赋的值会做相应的处理。在定义函数和程序设计中也允许对变量进行类型说明。这部分内容请看第 7 章。

1.2.2　给变量赋值

在 Maple 中，运算符"：="起赋值作用，一般形式为：

变量名：= 表达式

先计算赋值号右边的表达式的值，再将计算结果赋给左边的变量。如果赋值语句以冒号结尾则不显示计算结果；以分号结尾(工作表模式)或空白(文件模式)则显示计算结果。

如果要清除已被赋值的变量，可用 unassign 函数：

unassign(变量名)

如果要对多个变量同时赋值，可将变量序列和表达式序列依次对应在赋值号两边，或在一行中放多个赋值语句，用分号或冒号相隔。

变量 1，…，变量 k：= 表达式 1，…，表达式 k

如果要查看变量是否已被赋值，可用 assigned 函数，运行结果为 true 或 false。

assigned(变量名)

【例 7】　变量赋值。

```
> a,b,c := 11,22,33;
```

$$a,b,c := 11,22,33$$

```
> x:=11:y:=22:z:=33:
> (a,b,c):=(b,c,a); # 3 个变量的数值交换
```

$$a,b,c := 22,33,11$$

```
> A:=Matrix([[1,2],[3,4]]);
```

$$A := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

一个表达式可占一行或多行。按 shift+Enter 键实现续行。

```
> abcde：=1122334455667788999988776655443322 11
          +sin(12345);
```

$abcde$ ：=112233445566778899998877665544332211+sin(12345)

1.2.3 变量替换

变量替换的一般形式为：

subs(变量 1 = 变量 2，表达式)
subs(等式 1,…,等式 k，表达式)

【例 8】 变量和表达式替换。

```
> subs(x=1,x+1);
```

2

```
> subs(a=1+x,b=2-y,(a+b+c)^2);
```

$(x+3-y+c)^2$

```
> subs(a+b=1,sin(a+b+c)); # subs 的功力不够,需要用代数替换 algsubs
```

$\sin(a+b+c)$

```
> algsubs(a+b=1,sin(a+b+c));
```

$\sin(1+c)$

1.3 函数和调用

1.3.1 常用初等数学函数

在任何一种高级语言中,都有 abs(x)、sqrt(x)和 sin(x)等初等数学函数,调用时实在参数 x 只能是数。在符号计算语言中,初等函数的实在参数可以是表达式,即可为整数、小数、复数、符号和带运算符号的表达式,计算结果为数值或函数表达式。本小节中列出部分初等数学函数的一般用法(表 1-3),并以数值计算为侧重点,更详细的内容请看 2.5.1 小节。

表 1-3　常用初等数学函数

函　数	说　明
abs(z)	实数绝对值或复数的模
argument(z)	复数辐角
conjuagte(z)	共轭复数
Re(z),Im(z)	复数的实部、虚部
factorial(n)	阶乘 $n!$
GAMMA(x)	Gamma 函数 $\Gamma(x)$
x!	阶乘或 Gamma 函数 $\Gamma(x+1)$
x^y	x^y
exp(x)	e^x
ln(x)或 log(x)	自然对数
log10(x)	常用对数
log[b](x)	以 b 为底的对数
sqrt(x)	平方根
ceil(x)	不小于 x 的最小整数
floor(x)	不大于 x 的最大整数
round(x)	四舍五入取整
trunc(x)	取 x 的整数部分
frac(x)	取 x 的小数部分
gcd(a,b)	整数或多项式的最大公因式
lcm(a1,a2,…,an)	整数或多项式的最小公倍式
max(a1,a2,…,an)	选取最大值
min(a1,a2,…,an)	选取最小值
rand()	随机整数
randomize()	初始化随机数生成
sin(x),cos(x),tan(x), csc(x),sec(x),cot(x)	三角函数
arcsin(x),arccsc(x),arccos(x), arcsec(x),arctan(x),arccot(x)	反三角函数

续表

函　　数	说　　明
sinh(x),cosh(x),tanh(x), csch(x),sech(x),coth(x)	双曲函数
arcsinh(x),arccsch(x),arccosh(x), arcsech(x),arctanh(x),arccoth(x)	反双曲函数

【例 9】 初等数学函数。

计算 135,279 和 468 的最大公约数和最小公倍数。

```
> gcd(gcd(135,279),468),lcm(135,279,468);
                    9,217620
```

计算 $\sqrt{1.2+3.4i}$。

```
> sqrt(sin(1.1)+2.2*I);
              1.277666818+0.8609443283I
> round(8/3),ceil(8/3),frac(8/3);
                    3, 3, 2/3
> floor(8/3),trunc(8/3);
                      2,2
> trunc(6.1),trunc(6.5),trunc(-6.2),trunc(-6.9);
                   6,6,-6,-6
> trunc(5.6+7.8*I),round(5.6+7.8*I); # 分别对实部和虚部调用函数
                   5+7I,6+8I
> seq(rand(),i=1..4);
    773012980023,730616292946,106507053657,396412723003
```

1.3.2 自定义函数和调用

1. 函数定义

本节只简单介绍函数定义和调用,关于函数定义和调用的更详细内容,请看第 7 章。

箭头运算符“->”是定义函数的专用运算符,定义函数的一般形式:

函数名 := 变量(或变量序列) -> 表达式

注：变量或变量序列相当于高级语言子程序中的形式参数。

【例 10】 定义函数 $f(x)=x^2+1$，并计算 $f(2)$。

```
> f := x-> x^2+1;
```

$$f := x \rightarrow x^2+1$$

```
> f(2);
```

$$5$$

定义二元函数 $g(x,y)=x-y+xy$，计算 $g(3,2)$。

```
> g := (x,y) -> x-y + x*y;
```

$$g := (x,y) \rightarrow x-y+xy$$

```
> g(3,2);
```

$$7$$

```
> h := x -> [x+1,2*x, x^3]; # 定义的函数值可以是一列表
```

$$h := x \rightarrow [x+1, 2x, x^3]$$

```
> h(5);
```

$$6,10,125$$

```
> p :=x-1;
```

$$p := x-1$$

```
> ff := unapply(p,x);
```

$$ff := x \rightarrow x-1$$

```
> ff(10);
```

$$9$$

```
> s :=(x,y,z) -> (x+y, y-z);
```

$$s := (x,y,z) \rightarrow x+y, y-z$$

```
> s(6,3,1);
```

$$9,2$$

2. 函数调用

在调用函数时，将形式参数换成实在参数。系统函数和用户定义的函数都被允许嵌套调用。map 语句也起着函数调用的作用。

【例 11】 调用函数。

```
> f :=x-> x^2;
```

$$f := x \rightarrow x^2$$

```
> g :=(x,y)-> x+y;
```

$$g := (x,y) \rightarrow x+y$$

```
> f(f(2)); # 嵌套调用
```

$$16$$

```
> g(f(2),g(3,f(4))); # 多层嵌套调用
```

$$23$$

【例 12】 用 map 语句调用函数。

```
> map(f,z);
```

$$z^2$$

```
> map(g,3,4);
```

$$7$$

```
> map(x -> x^2, x+y);
```

$$x^2+y^2$$

```
> map(h,y*z);
```

$$h(y)h(z)$$

```
> map(h,{a,b,c});
```

$$\{h(a),h(b),h(c)\}$$

```
> map(h,[a,b,c],x,y);
```

$$[h(a,x,y),h(b,x,y),h(c,x,y)]$$

1.4 构造性数据类型

1.4.1 表达式序列(exprseq)

表达式序列(简称序列)是 Maple 的一个数据类型。序列的元素用逗号操作符连接,可以直接将序列赋值到一个变量中。逗号操作符是所有操作符中优先级别最低的操作符。我们会在多种数据类型中见到它。序列的一般形式为:

表达式 1,表达式 2,…,表达式 n

可用函数 seq 构造数值或函数序列。例如,

```
seq(f(i),i=1..n)     # 生成序列 f(1),f(2),…,f(n)
seq(f(i),i=m..n)     # 生成序列 f(m),f(m+1),…,f(n)
```

命令 seq(f(i),i=x)中的 x 可为集合(set)或列表(list)。

【例 13】 赋值或生成序列。

```
> A:=1,2,3; B:=4,5,6;
```

$$A := 1,2,3$$

$$B := 4,5,6$$

```
> s := A,a,b,B; # 逗号操作符组合生成新序列
```

$$s := 1,2,3,a,b,4,5,6$$

```
> X := seq(i^2, i=1..5); # i 的值从 1 到 5
```

$$X := 1,4,9,16,25$$

用"[]"操作符访问序列的元素,序列的下标从 1 开始。

```
> X[3];
```

$$9$$

```
> seq(i+1,i=X); # 定义域是表达式序列
```

$$2,5,10,17,26$$

```
> seq(sin(Pi*i/6), i=0..6);
```

$$0,\ \frac{1}{2},\ \frac{1}{2}\sqrt{3},\ 1,\ \frac{1}{2}\sqrt{3},\ \frac{1}{2},\ 0$$

```
> seq(x[i], i=1..5);
```

$$x_1,\ x_2,\ x_3,\ x_4,\ x_5$$

```
> seq(i, i="a".."f");
```

"*a*","*b*","*c*","*d*","*e*","*f*"

有时 \$ 出现在序列的运算中,\$ 也是构造序列的运算符,expr\$n 表示生成 n 个 expr 的表达式。

【例 14】 \$ 运算符。

```
> A := $2..5; B := 1,2,3$4;
```

$$A := 2,3,4,5$$

$$B := 1,2,3,3,3,3$$

```
> whattype(A);
```

exprseq

```
> seq(diff(x^5,x$n),n=1..4); # 计算 x^5 的 1 至 4 阶导数
```

$$5x^4,\ 20x^3,\ 60x^2,\ 120x$$

```
> add(diff(x^5,x$n),n=1..4); # 对序列所有元素求和
```

$$5x^4+20x^3+60x^2+120x$$

```
> mul(sin(n),n=2..4); # 求序列所有元素乘积
```

sin(2)sin(3)sin(4)

1.4.2 列表(list)

列表是用方括号括起来的表达式序列,序列常作为列表的元素,序列和列表的区

别在于有没有方括号。列表的元素以逗号相隔,元素的位置是有序号的,不同序号元素的值可以相同,列表的元素还可为列表。列表定义形式:

[表达式 1,表达式 2,…,表达式 n]

【例 15】 列表的定义和访问。

```
> La := [seq(i^2,i=1..5)];
                    [1,4,9,16,25]
> whattype(La);
                        list
> L := [1,[2,3],[4,[5,6],7],8,9];
               [1,[2,3],[4,[5,6],7],8,9]
```

要访问列表可用 op 命令或"[]"操作符。

```
> op(L);
                1,[2,3],[4,[5,6],7],8,9
> op(3,L); # 访问 L 的第 3 个元素
                    [4,[5,6],7]
> op(2..3,L); # 访问 L 的第 2 个到第 3 个元素
                  [2,3],[4,[5,6],7]
> L[2],L[2,1],L[3,2,1];
                     [2,3],2,5
```

L[3,2,1]表示取列表 L 的第 3 个元素的第 2 个元素的第 1 个元素。

```
> L[-1],L[-2]; # L 的倒数第 1、第 2 个元素
                        9,8
> L := [seq(x[i],i=1..4)];
```

$$L:=[x_1,\ x_2,\ x_3,\ x_4]$$

```
> L[2];
```

$$x_2$$

```
> solve([x-y^2=-2,x+y=0]); # 列表的元素是方程
```

$$\{y=-2, x=-2\}, \{y=1, x=-1\}$$

1.4.3 集合(set)

在 Maple 中 set 表示数学上的集合,集合的元素互不相同。集合中的每个元素都是唯一的,元素之间用逗号分隔,用花括号"{}"括起。集合的元素按某种次序排列,通过下标访问指定位置的元素。注意:元素的位置会随着操作而改变。集合定义

形式：

{元素 1,元素 2,…,元素 n}

常用的集合运算函数见表 1-4。

表 1-4　集合运算函数

函　　数	说　　明
A union B	$A \cup B$
A insert B	$A \cap B$
A minus B	$A \cap B^C$
A subset B	判断是否 $A \subseteq B$
member(a,B)	判断是否 $a \in B$
evalb(A=B)	判断是否 $A=B$
A[i] 或 op(i,A)	A 中第 i 个元素

【例 16】 集合运算。

```
> A：={a,2,a,t,1,2}；# 去掉重复元素并重新排序
```

$$\{1,2,a,t\}$$

```
> B：={t,c,2,3}；
```

$$\{2,3,t,c\}$$

```
> A union B；# 计算集合 A 和集合 B 的和集
```

$$\{1,2,3,a,t,c\}$$

```
> A minus B；# 在集合 A 而不在集合 B 中的元素集合
```

$$\{1,a\}$$

```
> B minus A；# 在集合 B 而不在集合 A 中的元素集合
```

$$\{3,c\}$$

```
> A subset B；
```

false

```
> convert([a,b,c,d],'set')；# 将列表类型转换为集合类型
```

$$\{a,b,c,d\}$$

1.4.4　向量和矩阵

有关向量和矩阵的详细定义请看第 4 章。

对于维数较小并具有值的向量和矩阵可以直接用尖括号构造，用〈|〉和〈,〉分

别构造行向量和列向量。请注意元素分隔符“,”和“|”的区别和作用。通俗地说，元素遇到分隔符“|”向前(右)走，遇到分隔符“,”向下走。例如：

```
> <1,2,3> ; # 构造列向量
```

$$\begin{bmatrix}1\\2\\3\end{bmatrix}$$

```
> <a|b|c> ; # 构造行向量
```

$$[a,\ b,\ c]$$

```
> <<1,2,3>|<4,5,6>> ;
```

$$\begin{bmatrix}1 & 4\\2 & 5\\3 & 6\end{bmatrix}$$

```
> <<1|2|3>,<4|5|6>> ;
```

$$\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}$$

思考：想一想下列命令的运行结果：

```
<<1|2|3>|<4|5|6>>
<<1|2|3>,<4|5|6>,<7|8|9>>
<<1,2,3>,<4,5,6>>
<<1,2,3>|<4,5,6>|<7,8,9>>
```

【例 17】 矩阵和列表数据结构的转换。

```
> A := Matrix(2,2,[[x,y],[z,o]]);
```

$$A := \begin{bmatrix}x & y\\z & o\end{bmatrix}$$

```
> convert(A,'list');
```

Error, (in convert/list) cannot convert rtable with more than one dimension having range greater than 0; try convert(A,listlist)

```
> convert(A,'listlist');
```

$$[[x,y],[z,o]]$$

```
> convert(A,'set');
```

$$\{x,y,z,o\}$$

1.5　表　达　式

1.5.1　算术运算符和表达式

算术表达式由常数、变量、函数、算术运算符和括号组成。常数和变量的类型有：整数、有理数、实数、复数、列表、向量和矩阵等。函数包括系统定义的函数、用户定义的函数和函数包中的函数。

表 1-5 所示为 Maple 中的算术运算符，算术运算的优先级遵守数学中的习惯：同级运算符遵守从左到右的顺序。

表 1-5　算术运算符

符　号	说　明	示　例
^	乘方	2^3
*	乘号	s * t
/	除号	m/n
+	加号	a+b
−	减号或负号	x−y, −u
%	上一个计算结果	
%%	倒数第二个计算结果	
%%%	倒数第三个计算结果	

1.5.2　逻辑表达式和逻辑运算符

关系表达式是最简单的逻辑表达式。用关系表达式表示一个判别条件，如 $x>3$。关系表达式的一般形式：

表达式 1　关系运算符　表达式 2

其中表达式可为数值表达式、字符表达式和意义更广泛的表达式，如图形表达式。在实际使用中，关系运算符（表 1-6）两边常常是数值表达式或字符表达式。

表 1-6　关系运算符

关系运算符	数学符号和意义	示　例
=	等于	3x=y−1
!=	不等于	x!=y
>	大于	x>y, x>y>z
>=	大于等于	x+y>=y/z
<	小于	u<v
<=	小于等于	y/s<=z

用一个关系表达式只能表示一个判定条件，要表示几个判定条件的组合必须用逻辑运算符将关系表达式组在一起。我们称表示判定条件的表达式为逻辑表达式。

逻辑表达式的一般形式为：

关系表达式 1　逻辑运算符　关系表达式 2

常用的逻辑运算符有：!（逻辑非），&&（逻辑与、and），||（逻辑或、or），见表 1-7。

表 1-7　逻辑运算符

逻辑运算符	说　　明
!	逻辑非。A 是真，则!A 是假；A 是假，则!A 是真
&&	逻辑与。仅当 A 和 B 都是真时，A&&B 的值是真
||	逻辑或。当 A 或 B 是真时，A||B 的值是真

逻辑表达式取值：真(true)和假(false)。例如：

```
> x:=1;y:=2; x<y;
                              1 < 2
> evalb(x<y); # evalb是计算逻辑表达式函数
                              true
> x>0 and y>0;
                              true
```

1.6　有关表达式计算函数

1.6.1　访问表达式成员

indets(表达式)	表达式的所有变量列表
indets(表达式,类型)	表达式中具有指定类型的变量列表
nops(表达式)	表达式的元素个数
op(i，表达式)	表达式的第 i 个元素
subsop(i=表达式 1，表达式 2)	将表达式 2 的第 i 个元素替换为表达式 1

【例 18】 访问表达式成员。

```
> expr：=sin(x) * x^2-cos(y+x)+3;
```

$$expr := \sin(x)x^2-\cos(y+x)+3$$

```
> nops(expr);
```

$$3$$

```
> op(2..3,expr); # expr 的第 2~3 个元素
```

$$-\cos(y+x),3$$

```
> op(1,op(1,expr)),op(1,op(2,expr));
```

$$\sin(x),-1$$

```
> subsop(2=0,expr); # 将 expr 的第 2 个元素置为零
```

$$\sin(x)x^2+3$$

```
> expr;
```

$$\sin(x)x^2-\cos(y+x)+3$$

```
> subsop(1=a,2=b,3=c,expr);
```

$$a+b+c$$

```
> subsop(0=g,f[a,b,c]),subsop(1=g,f[a,b,c]);
```

$$g_{a,b,c},f_{a,b,c}$$

1.6.2　计算函数 eval 和 value

在 eval 后面加以不同的后缀表明不同的计算对象和计算要求。value 主要计算惰性函数的值。

eval(表达式)	计算表达式的值
eval(表达式,x=a)	计算表达式在 $x=a$ 时的值
evalf(表达式)	计算表达式的浮点值
evalf[n](表达式)	计算表达式的 n 位有效数字
evalhf(表达式)	硬件计算表达式的浮点值
evalb(表达式)	计算逻辑表达式的值
evalm(表达式)	计算矩阵表达式的值
value(表达式)	计算惰性函数的值

【例 19】 表达式求值。

```
> poly:=x^3+3*x+2;
```

$$poly:=x^3+3x+2$$

```
> eval(poly,x=1);
```

$$6$$

```
> subs(x=1,poly);
```

$$6$$

```
> F:=Int(n*x^(n-1),x);
```

$$F:=\int nx^{(n-1)}\,dx$$

```
> value(F);
```

$$x^n$$

```
> evalf[20](3+cos(2)+sin(5)*I); # 保留 20 位有效数字
```

$$2.5838531634528576130-0.95892427466313846889I$$

1.6.3 判断表达式类型函数

is(表达式,属性)	判断表达式是否具有所述属性
type(表达式,类型)	判断表达式是否具有所述类型
whattype(表达式)	给出表达式的类型

其中表达式类型有 *(乘法),+(加法),.(点乘),..(区间),::(类型匹配),<(小于),<=(小于等于),<>(不等于),=(等于),^(指数),and(逻辑与),array(数组),complex(复数),exprseq(表达式序列),extended_numeric(广义数值),float(实数),fraction(分数),function(函数),implies(逻辑蕴含),indexed(指标),integer(整数),list(列表),module(模块),moduledefinition(模块定义),not(逻辑非),or(逻辑或),polynom(多项式),procedure(过程),series(序列),set(集合),string(字符串),

symbol(符号),table(表格),uneval(延迟赋值),unknown(未知类型),xor(逻辑异或),Array(数组),Matrix(矩阵),Vector[column](列向量),Vector[row](行向量)等。

习　题

1. 计算下列各式的数值。

(1) $12 \cdot 2^{2006}$　　(2) e^{6-8i}

(3) $\sin 15^\circ + \cos 15^\circ$　　(4) $\log_5 2475$

(5) $\ln(1+e^{-2})$　　(6) $\sqrt{|\ln\cos(75^\circ)|}$

(7) $\cos\left(2\arccos\frac{1}{3} - \arccos\frac{1}{6}\right)$　　(8) $\tan\left(\arctan\frac{\sqrt{2}}{2} + \arctan\frac{\sqrt{2}}{3}\right)$

(9) $12\left(\cos\frac{3\pi}{2} + i\sin\frac{3\pi}{2}\right) \div 4\left(\cos\frac{\pi}{3} + i\sin\frac{\pi}{3}\right)$

2. 计算 761,1602,3115 的最大公约数。

3. 计算 49,102,147 的最小公倍数。

4. 分别用 subs、eval 和 value 计算 $f(x,y)=\sin x+y^2+|xy|$,$x=1.2$,$y=0.7$。

5. 建立下列数据序列,并对所有序列元素求和。

(1) $\frac{1}{2}, \frac{1}{3}, \cdots, \frac{1}{100}$

(2) $\frac{1+i}{1+i^4}$, $i=1,2,\cdots,20$

(3) $(2n-1)^3$, $n=1,2,\cdots,20$

6. 建立表格 $\begin{pmatrix} 11 & 12 & 13 & 14 \\ 21 & 22 & 23 & 24 \\ 31 & 32 & 33 & 34 \\ 41 & 42 & 43 & 44 \end{pmatrix}$。

第2章 初等数学

2.1 多项式运算

多项式是初等数学的基本元素，在高等数学中依然扮演重要的角色，如多项式插值、多项式逼近等。在 Maple 中，多项式是由数字、常量、变量和算术运算符+、-、*、/、^组成的表达式。

多项式分为单变量多项式和多变量多项式。单变量多项式是只包含一个变量的多项式，多变量多项式包含两个及两个以上的变量。多项式的类型名为 polynom，多项式中未知变量的指数必须是非负整数。

多项式是一类特殊形式的表达式，表达式中的各种运算都可用在多项式的运算中。同样，多项式的运算函数也能用在一般表达式中。例如，表达式的各种输出形式可用于多项式的输出。

【例 1】 多项式输出。

```
> p := (x+10)^5+(x-5)^3+87;
```

$$p := (x+10)^5+(x-5)^3+87$$

```
> type(p,polynom); # p是否是多项式数据类型
```

true

可以用命令 indets 查看多项式中的未知变量，返回结果以集合形式表示。

```
> p := 3*x*y+x-y+1+x^3*y^2;
> indets(p);
```

$$\{x, y\}$$

```
> sort(p); # 多项式各项由高到低按降幂排列，与sort(p,descending)等效
```

$$x^3 y^2+3xy+x-y+1$$

```
> sort(p,[x,y],ascending); # 按x,y的升序排列
```

$$1-y+x+3xy+x^3 y^2$$

```
> sort(p,[y,x],ascending); # 按y,x的升序排列
```

$$1+x-y+3yx+y^2x^3$$

```
> subs({x=a,y=2},p); # 表达式求值的简单方式
```

$$-1+7a+4a^3$$

设置选项 horner，将多项式转换为乘法次数最少的 horner 多项式形式。

```
> convert(1-x+2*x^2-3*x^3+4*x^4,horner);
```

$$1+(-1+(2+(-3+4x)x)x)x$$

2.1.1　多项式的分解和展开

常用的化简多项式的命令有 expand(展开)、factor(因式分解)、collect(合并同类项)和 simplify(化简)等。这些命令不仅用于多项式化简，也可以用于其他表达式的化简。

1. 多项式展开 expand

expand(表达式)	展开表达式为单项式之和
expand(表达式,exp1,…,expn)	按因式 exp1,…,expn 展开表达式

expand 能够对三角函数、对数函数、指数函数、双曲函数、分段函数等各类表达式进行展开，并根据三角函数关系对三角运算的表达式进行展开和化简。

【例 2】　表达式展开。

```
> expand((x+1)*(y+2)*(z+3)); # 多项式展开
```

$$yxz+3yx+2xz+6x+yz+3y+2z+6$$

```
> expand((x+1)*sin(x+y)); # 展开含有三角函数的表达式
```

$$x\sin(x)\cos(y)+x\cos(x)\sin(y)+\sin(x)\cos(y)+\cos(x)\sin(y)$$

```
> expand((x+1)/(x+2)); # 有理分式展开
```

$$\frac{x}{x+2}+\frac{1}{x+2}$$

在因式展开过程中保持不变的子表达式。

```
> expand((x+1)*(y+z),x+1);
```

$$(x+1)y+(x+1)z$$

```
> expand((x+1)*(y+z)*sin(x+y),x+1,sin(x+y));
```

$$(x+1)\sin(x+y)y+(x+1)\sin(x+y)z$$

2. 因式分解 factor

factor(表达式,数域)

在数域中，对表达式作因式分解，数域可以是 real、complex 或用户自定义的数域，在缺省情况下，Maple 会根据所给多项式的系数确定因式分解的域。例如，如果所给多项式的系数都是整数，命令 factor 分解后的每个因式都是具有整数系数的不可再分因式，其中的因式可能不是线性的。

factor 不能对整数进行因式分解，如果要对整数进行因式分解，可以使用命令 ifactor。例如：

```
> ifactor(1260);
```

$$(2)^2(3)^2(5)(7)$$

【例 3】 观察表达式在不同数域的分解效果。

```
> factor(x^3+3*x^2-25*x+21);
```

$$(x-1)(x-3)(x+7)$$

```
> factor(x^3+5); # 默认数域为有理数域
```

$$x^3+5$$

```
> factor(x^3+5,real); # 在实数域中
```

$$(x+1.709975947)(x^2-1.709975947x+2.924017740)$$

```
> factor(x^3+5,complex); # 在复数域中
```

$$(x+1.709975947)(x-0.8549879733+1.480882610I)$$
$$(x-0.8549879733-1.480882610I)$$

```
> factor((x^2-3)*(x^2-5),{sqrt(3),sqrt(5)});
```

$$(x-\sqrt{5})(x+\sqrt{3})(x-\sqrt{3})(x+\sqrt{5})$$

3. 合并同类项 collect

collect(表达式,变量)
collect(表达式,变量,规则)

其中表达式为任意的代数表达式，通常为多项式，变量可以是单一的，也可以是变量列表或集合。

collect 将多项式中所有具有相同有理数幂的项合并在一起，以多种组合方式书写多变量多项式。collect 没有排序功能，用户常常误认为它应当包含排序功能。

【例 4】 按变量次序合并同类项。

```
> p:=x*y+a*x*y+y*x^2-a*y*x^2+x+a*x;
```

$$p:=yx+axy+yx^2-ayx^2+x+ax$$

```
> collect(p,x); # 按 x 为变量合并
```

$$(y-ay)x^2+(y+ay+1+a)x$$

```
> collect(p,y); # 按 y 为变量合并
```

$$(x^2-ax^2+x+ax)y+x+ax$$

```
> collect(p,[y,x]);
```

$$((1-a)x^2+(1+a)x)y+(1+a)x$$

【例5】 按函数(表达式)合并同类项。

```
> f:=a*ln(x)-ln(x)*x-x; collect(f,ln(x));
```

$$(a-x)\ln(x)-x$$

```
> g:=int(x*(exp(x)+exp(-x)),x); collect(g,exp(x));
```

$$(x-1)e^x+\frac{-1-x}{e^x}$$

```
> f:=a^3*x-x+a*x^3+a;
```

$$f:=a^3x-x+ax^3+a$$

```
> collect(f,x);
```

$$ax^3+(a^3-1)x+a$$

```
> collect(f,x,factor); # 选项 factor 的作用
```

$$ax^3+(a-1)(a^2+a+1)x+a$$

【例6】 观察 collect 与 sort 的区别。

```
> collect(x^3*y+(x-y)^2+3+4*x,y);
```

$$y^2+(-2x+x^3)y+4x+x^2+3$$

```
> sort(%);
```

$$x^2+y^2+4x+(x^3-2x)y+3$$

4. 化简多项式 simplify

simplify 可按因式或数学规则化简表达式，如三角函数规则，或者用户自定义规则。

simplify(表达式)	化简表达式
simplify(表达式,规则)	按照规则化简表达式
simplify(表达式,assume=性质)	按给定的性质化简表达式
simplify(表达式,symbolic)	对表达式进行符号处理

【例7】 化简表达式。

```
> f:=sqrt(x^2); simplify(f);
```

$$\text{csgn}(x)\ x$$

```
> simplify(f,assume=positive);
```

$$x$$

```
> simplify(f,assume=real);
```

$$|x|$$

```
> simplify(f,symbolic);
```

$$x$$

```
> f:=x^3*y^2+(x+y)^2*x-(2*y*x^2+x*y^2):
> simplify(f); # 化简多项式
```

$$x^3y^2+x^3$$

```
> simplify(f,{x^2=x+1}); # 花括号{ }不能少
```

$$y^2+1+(2y^2+2)x$$

```
> f:=-1/3*x^5*y+x^4*y^2+1/3*x*y^3+1:
> simplify(f,{x^3=x*y,y^2=x+1});
```

$$1+y^5+y^4-2y^3-y^2+y$$

【例 8】 化简三角函数表达式

```
> simplify(cos(x)^5+sin(x)^4+2*cos(x)^2-2*sin(x)^2-cos(2*x));
```

$$\cos(x)^4(\cos(x)+1)$$

2.1.2 多项式的四则运算

1. 加法和减法

用加号“+”和减号“-”运算符完成多项式的加法和减法运算。常用函数 sort 整理多项式，让各项按幂次从高到低排列。

【例 9】 多项式的加法。

```
> f:=x^2-x+1: g:=x^3+2*x^2-3: f+g;
```

$$3x^2-x-2+x^3$$

```
> simplify(%);
```

$$3x^2-x-2+x^3$$

```
> sort(%); # 等同 sort(%,descending)
```

$$x^3+3x^2-x-2$$

2. 乘法

两个多项式相乘，自然想到在两个多项式之间用乘号，不过仅用乘号还不够，还要用 expand 或 factor 告诉 Maple 多项式相乘后是展开形式还是因式形式。

【例 10】　多项式的乘法。

```
> f:=x^2-x+1; g:=x^3+2*x^2-3; f*g; # 为何乘号不起作用
```

$$(x^2-x+1)(x^3+2x^2-3)$$

```
> expand(f*g); # f与g的乘积按展开形式表示
```

$$x^5+x^4-x^2-x^3+3x-3$$

```
> factor(f*g); # f与g的乘积按因式形式表示
```

$$(x^2-x+1)(x-1)(x^2+3x+3)$$

3. 除法

用函数 rem 或 quo 做多项式的除法。当多项式的系数都是有理数时，函数 divide也能做多项式的除法。设 f,g 是以 x 为自变量的一元多项式，a,b 为任意有理系数多项式。

rem(f,g,x)	计算 f/g 的余式
rem(f,g,x,'q')	计算 f/g 的余式，并将商赋给变量 q
quo(f,g,x)	计算 f/g 的商
quo(f,g,x,'r')	计算 f/g 的商，并将余式赋给变量 r
divide(a,b,'q')	判断 a 是否整除 b。若整除，将商赋给变量 q

【例 11】　多项式的除法。

```
> f:=x^5-x+1: g:=x^2-x+1:
> r:=rem(f,g,x,'q'); # 计算f/g的余式，将f/g的商放在q中
```

$$r:=2-2x$$

```
> quo(f,g,'x'),q; # 计算f/g的商
```

$$x^3+x^2-1, x^3+x^2-1$$

```
> simplify(g*q+r);
```

$$x^5-x+1$$

```
> gcd(x^2-y^2,x^5-y^5); # 计算f和g的最大公因式
```

$$-y+x$$

```
> lcm(x^2-y^2,x^5-y^5); # 计算f和g的最小公倍式
```

$$(x+y)(x^5-y^5)$$

```
> divide(x^3-y^3,x-y,q);
```

true

```
> q;
```

$$x^2+xy+y^2$$

```
> sort(%);
```

$$x^2+xy+y^2$$

4. 判别式 discrim

我们熟知二项式 ax^2+bx+c 的判别式为 $\Delta=b^2-4ac$。对于一般的 n 次多项式 $p(x)=a_n(x-x_1)\cdots(x-x_n)$，其判别式定义为 $\Delta=a_n^{n-1}\prod_{1\leqslant i<j\leqslant n}(x_i-x_j)^2$。

【例 12】 二项式和三项式的判别式。

```
> discrim(a*x^2+b*x+c,x);
```

$$-4ac+b^2$$

```
> discrim(a*x^3+b*x+c,x);
```

$$-a(4b^3+27c^2a)$$

```
> discrim(a*x^3+b*x^2+c*x+d,x);
```

$$-27a^2d^2+18adcb+b^2c^2-4b^3d-4ac^3$$

2.1.3 访问多项式的部分元素

对于一个项数较多的多项式，在运算中有时想查看某个变量的某幂次项的系数，或查看多项式中的最高幂次，或替换多项式中的某一项。在表 2-1 中，设 n 为整数，p 为多项式，x 为单个变量或变量列表或变量集合。

表 2-1 访问多项式的部分元素

函 数	说 明
indets(p)	p 的所有变量列表
coeff(p,x,n) coeff(p,x^n)	p 的 x^n 项系数，n 的缺省值为 1
coeffs(p)	p 的所有单项式的系数列表
coeffs(p,x)	p 关于 x 的多项式系数列表
coeffs(p,x,'t')	p 关于 x 的多项式系数列表，并将相应的 x 的幂次赋给变量 t
lcoeff(p)	p 的首项系数
lcoeff(p,x)	p 关于 x 的最高项系数
loceff(p,x,'t')	p 关于 x 的最高项系数，并将相应的 x 的幂次赋给变量 t
tcoeff(p)	p 的尾项系数
tcoeff(p,x)	p 关于 x 的最低项系数
toceff(p,x,'t')	p 关于 x 的最低项系数，并将相应的 x 的幂次赋给变量 t

续表

函 数	说 明
degree(p,x)	p 关于 x 的最高项的全次数（当 x 为单个变量或集合时）或向量次数（当 x 为列表时）
ldegree(p,x)	p 关于 x 的最低项的全次数（当 x 为单个变量或集合时）或向量次数（当 x 为列表时）

【例 13】 访问多项式的部件。

```
> dxs:=expand((x*y-2*y^2-1)^3);
```

$$dxs := x^3y^3 - 6x^2y^4 - 3x^2y^2 + 12xy^5 + 12xy^3 + 3xy - 8y^6 - 12y^4 - 6y^2 - 1$$

```
> nops(dxs); # 共有 10 项元素
```

$$10$$

```
> degree(dxs); # 给出幂次最高项的系数
```

$$6$$

```
> coeff(dxs,y,2); # 列出 y^2 项的系数
```

$$-3x^2 - 6$$

```
> lcoeff(dxs),tcoeff(dxs); # 访问首项、尾项系数
```

$$1, -1$$

```
> op(3,dxs); # 访问第 3 个元素
```

$$-3x^2y^2$$

2.2 有理分式

2.2.1 有理分式的类型

多项式的加、减、乘和非负整数次乘方运算的结果仍然是多项式，两个多项式的商可能是分式而不是多项式，Maple 把多项式和分式统称为有理多项式或有理分式，其中多项式可以看成分母为 1 的分式。

函数 expand 只能对分式的分子进行展开，并把结果表示为多个分式和的形式；函数 factor 首先把分式化简，然后对分式的分子和分母进行因式分解；函数 collect 对分式通常不起作用。

【例 14】 有理分式的展开和分解。

```
> f:=(x+1)^2/((x^2+x)*x); g:=expand(f);
```

$$g:=\frac{x}{x^2+x}+\frac{2}{x^2+x}+\frac{1}{(x^2+x)x}$$

```
> factor(f);
```

$$\frac{x+1}{x^2}$$

有理分式的类型名为 ratpoly,可以使用命令 type 确认有理分式的数据类型。

```
> p:=x^3+4*x^2+5*x+2: q:=x^2+3*x+2: rp:=p/q;
```

$$rp:=\frac{x^3+4x^2+5x+2}{x^2+3x+2}$$

```
> [whattype(rp),type(rp,ratpoly),type(rp,polynom)];
```

['*',true,false]

有理分式包含两个操作数:分母和分子。可以使用命令 op(1,rp)访问有理分式 rp 的分子,op(2,rp)访问 rp 的分母。

numer(表达式)	表达式的分子
denom(表达式)	表达式的分母

【例 15】 观察用 op 与 denom 访问有理分式分母的效果。

```
> numer(rp),denom(rp); # 访问有理分式的分子和分母
```

$$x^3+4x^2+5x+2, x^2+3x+2$$

```
> op(1,rp),op(2,rp); # 访问有理分式的第 1、2 个操作数
```

$$x^3+4x^2+5x+2, \frac{1}{x^2+3x+2}$$

2.2.2 分式化简

normal(表达式)	将表达式化为标准分式

通常 normal 对多项式不起作用。simplify 可以化简分式。convert 有时也能起到化简分式的作用。

【例 16】 有理分式化简。

```
> f:=(x+1)^2/((x^2+x)*x): normal(f); # 此时与 simplify(rp)作用相同
```

$$\frac{x+1}{x^2}$$

```
> collect(f,x);
```

$$\frac{1+x^2+2x}{(x^2+x)x}$$

```
> convert(f,parfrac,x); # 化简效果不错
```

$$\frac{1}{x}+\frac{1}{x^2}$$

```
> convert(exp(x),confrac,x); # 将 e^x 化为连分式
```

$$1+\cfrac{x}{1+\cfrac{x}{-2+\cfrac{x}{-3+\cfrac{x}{2+\frac{1}{5}x}}}}$$

2.2.3 变量替换

分式的变量替换和多项式的变量替换类似，可以通过命令 subs 或 eval 完成变量替换。

【例 17】 变量替换 subs 求值。

```
> f:=(x^2+3*x+2)/(x^2+5*x+6):
> subs(x=1.0,f), eval(f,x=1.0);
                    0.5000000000, 0.5000000000
> subs(x=-2,f);
             Error, numeric exception: division by zero
```

错误信息表明替换命令不能在分式的奇点处进行变量替换。

2.3 求和与乘积

Maple 提供了计算求和与乘积的命令(表 2-2)，计算有限和式、无限和式、有限乘积以及无限乘积。

sum、product 的求和、乘积意义与 add、mul 基本相同，两组函数只在通项公式和运算范围等细节上存在差别。add 和 mul 要求循环范围是确定的数，更适合数值序列求和、求积；sum、product 的循环范围可以是文字或无穷，更适合做公式推导；add、mul 能够对数组元素进行运算，而 sum、product 则不能直接对数组元素进行运算。

另外，sum、product 还有相应的惰性函数 Sum、Product。

表 2-2 常用的求和与乘积函数

函 数	说 明
add(f,i=m..n), add(f,i=x),add(f,i in x)	数值求和 $\sum_{i=m}^{n} f(i)$, $\sum_{i\in X} f(i)$
mul(f,i=m..n), mul(f,i=x),mul(f,i in x)	数值乘积 $\prod_{i=m}^{n} f(i)$, $\prod_{i\in x} f(i)$
sum(f,i),sum(f,i=m..n)	形式求和 $\sum_{i} f(i)$, $\sum_{i=m}^{n} f(i)$
product(f,i),product(f,i=m..n)	形式乘积 $\prod_{i} f(i)$, $\prod_{i=m}^{n} f(i)$

【例 18】 观察 sum 和 add 对求和范围的反应。

```
> sum(k^2, k=1..n);
```

$$\frac{1}{3}(n+1)^3-\frac{1}{2}(n+1)^2+\frac{1}{6}n+\frac{1}{6}$$

```
> factor(%);
```

$$\frac{1}{6}n(n+1)(2n+1)$$

```
> add(k^2, k=1..n);
```

Error, unable to execute add

```
> Sum(1/k^2,k=1..infinity) = sum(1/k^2,k=1..infinity);
```

$$\sum_{k=1}^{\infty}\frac{1}{k^2}=\frac{1}{6}\pi^2$$

```
> add(1/k^2,k=1..infinity);
```

Error, unable to execute add

如果不定义 sum()中变量 k 的取值范围，则系统会自动选择从 1 到 $k-1$，如下：

```
> sum(k,k);
```

$$\frac{1}{2}k^2-\frac{1}{2}k$$

```
> sum(x^k,k=1..1000);
```

$$\frac{x^{1001}}{x-1}-\frac{x}{x-1}$$

```
> add(x^k,k=1..1000);
```

输出略，请在计算机上观看结果，输出求和式的每一项。

【例 19】 求和范围是列表类型。

```
> L：=[seq(i,i=1..10)]; add(i,i=L);
```

L：=[1,2,3,4,5,6,7,8,9,10]

55

```
> sum(i,i=L);
```

Error, (in sum) second argument must be a name, name＝a..b, name＝RootOf, or name＝value

【例 20】 下标变量与求和通项。

```
> v：= Vector([1,2,3,4,5]): add(v[k],k=1..5);
```

15

```
> sum(v[k],k=1..5);
```

Error, bad index into Vector

```
> sum('v[k]',k=1..5);
```

15

```
> w：= [1,2,3,4,5]: sum(w[k],k=1..5);
```

15

2.4　初等代数方程和方程组

2.4.1　方程及其根的表示

本节所涉及的方程和方程组属于初等数学的范围，有关高等数学求解线性方程组的内容放在第 4 章。

在数学中，代数方程用等号表示，例如：$x^2-6x-5=0$。在 Maple 中，代数方程同样使用逻辑等号"＝"表示，此时方程是一个逻辑表达式，它的意义是判断方程等号两边的量是否相等。

```
> x：=1: evalb(x^2-6*x-5=-10); # x是否为方程的根
```

true

```
> evalb(x^2-6*x-5=0); # false表明方程两边不相等
```

false

lhs(方程),rhs(方程)

分别获取方程的等号左边和右边的项。

【例 21】 访问方程组的左边(右边)。

```
> eqn := x^4-5*x^2+6*x=2: lhs(eqn),rhs(eqn);
```

$$x^4-5x^2+6x, 2$$

```
> eqnl := lhs(eqn)-rhs(eqn)/2=3; # 组合为新方程
```

$$eqnl := x^4-5x^2+6x-1=3$$

isolate(方程,表达式)

化简方程,使得表达式仅出现在方程的左边,右边不见其影。

【例 22】 用移项函数解方程。

```
> isolate(y+x*y+x^2=x-y,x);
```

$$x=-\frac{1}{2}y+\frac{1}{2}+\frac{1}{2}\sqrt{y^2-10y+1}$$

```
> isolate(y-sin(x)+x*cos(y)=2,cos(y));
```

$$\cos(y)=\frac{2-y+\sin(x)}{x}$$

```
> limit(f(x),x=infinity)-w+int(h(x),x)-int(g(y),y)=s(x);
```

$$\lim_{x\to\infty}f(x)-w+\int h(x)\mathrm{d}x-\int g(y)\mathrm{d}y=s(x)$$

```
> isolate(%,x);
```

$$\lim_{x\to\infty}f(x)+\int h(x)\mathrm{d}x-s(x)=w+\int g(y)\mathrm{d}y$$

2.4.2 求解方程(solve)

solve(方程,变量)

求解方程中的变量,方程可以是方程组或不等式(组),变量可以是变量列表或集合,还可以省略。当方程元素是符号或准确数时,solve 求方程的准确解(解析解);当方程元素是实数时,solve 计算方程的近似解(数值解)。方程的解按序列形式给出,可赋值于列表、集合等数据类型,并按序列元素访问方式访问每一个解。

【例 23】 观察解的数据类型。

```
> sol := solve(x^3-2*x+1=0,x); # 或 solve(x^3-2*x+1)
```

$$sol := 1, -\frac{1}{2}+\frac{1}{2}\sqrt{5}, -\frac{1}{2}-\frac{1}{2}\sqrt{5}$$

```
> sol[2]; # 访问第 2 个解,解集的第 2 个元素
```

$$-\frac{1}{2}+\frac{1}{2}\sqrt{5}$$

方程的解有多种数据类型，解的输出形式取决于对解序列的类型定义。

```
> sols：=solve(x^4-6*x^3+13*x^2-12x+4)；# 解的类型是序列
```

$$sols := 2,2,1,1$$

```
> sols[2..4]；# 可以像使用 sequence/list 一样
```

$$2,1,1$$

```
> {sols}；# {} 给出的是集合的形式，所以不能有重复的元素
```

$$\{1,2\}$$

```
> sols：=solve({x^2+y^2=5,x^2-y^2=3})；# 解的类型是序列
```

$$sols := \{x=-2,y=1\},\{x=2,y=1\},\{x=-2,y=-1\},\{x=2,y=-1\}$$

```
> sols：=solve({x^2+y^2=5,x^2-y^2=3},[x,y])；# 解的类型是列表
```

$$sols := [[x=-2,y=1],[x=2,y=1],[x=-2,y=-1],[x=2,y=-1]]$$

【例 24】 求解 $x^3+10x^2+7x-18=0$，并验证方程的根。

```
> f：= x^3+10*x^2+7*x-18；solve(f)
```

$$1,-2,-9$$

```
> eval(f,x=-2) # 验证-2 是否为方程的根
```

$$0$$

或者利用 subs()函数将求出的解代换回方程来检验。

```
> subs(x=-9,f)；# 验证-9 是否为方程的根
```

$$0$$

```
> subs(x=3,f)；# 验证 3 是否为方程的根
```

$$120$$

有时准确解过于复杂不便使用，如方程

```
> solve(x^3-12*x^2+3)；
```

$$\frac{1}{2}(500+4I\sqrt{759})^{(1/3)}+\frac{32}{(500+4I\sqrt{759})^{(1/3)}}$$

$$+4,-\frac{1}{4}(500+4I\sqrt{759})^{(1/3)}-\frac{16}{(500+4I\sqrt{759})^{(1/3)}}+4$$

$$+\frac{1}{2}I\sqrt{3}\left(\frac{1}{2}(500+4I\sqrt{759})^{(1/3)}-\frac{32}{(500+4I\sqrt{759})^{(1/3)}}\right),$$

$$-\frac{1}{4}(500+4I\sqrt{759})^{(1/3)}-\frac{16}{(500+4I\sqrt{759})^{(1/3)}}$$

$$+4-\frac{1}{2}I\sqrt{3}\left(\frac{1}{2}(500+4I\sqrt{759})^{(1/3)}-\frac{32}{(500+4I\sqrt{759})^{(1/3)}}\right)$$

其中 I 是虚数单位。这时，我们可以用 evalf()函数得到方程的近似解。

```
> evalf(%)；
```

$$11.97909389-1.10^{-10}I,-0.4900922165-1.732050808\ 10^{-9}I,$$
$$0.5109983305+1.732050808\ 10^{-9}I$$

如果将方程其中一个元素的整数系数改为小数(例如,在系数 3 后面加一个小数点),solve()函数直接给出方程的实数近似解。

```
> solve(x^3-12*x^2+3.);
```

$$0.5109983303,\ 11.97909389,\ -.4900922162$$

【例 25】 解的参数形式。

```
> solve(x^2+2*y^2=9,{x(t),y(t)});
```

$$\left\{x=\frac{3}{\sqrt{1+2t^2}},y=\frac{3t}{\sqrt{1+2t^2}}\right\},\left\{x=-\frac{3}{\sqrt{1+2t^2}},y=-\frac{3t}{\sqrt{1+2t^2}}\right\}$$

solve()函数也可以求非多项式方程的精确解,但如果方程过于复杂(如混合出现指数式、多项式、三角式),其精确解往往不易求得,此时求近似解是较佳的选择。

2.4.3 求解方程组

solve()函数主要用在求解低阶的线性方程组或多项式方程组,或者说求解初等数学中的方程组。在本节中方程组无解和无穷多组解的情况不是我们关注的内容。

【例 26】 解方程组 $\begin{cases}u+v+w=1\\3u+v=3\\u-2v-w=0\end{cases}$,并验证方程组的解。

```
> eqns:={u+v+w=1,3*u+v=3,u-2*v-w=0}; sols:=solve(eqns);
```

$$sols:=\left\{v=\frac{3}{5},w=\frac{-2}{5},u=\frac{4}{5}\right\}$$

```
> eval(eqns,%); # 验证方程组的解
```

$$\{1=1,0=0,3=3\}$$

还可以用 subs()函数来验证求得的解。

```
> subs(sols,eqns);
```

$$\{1=1,0=0,3=3\}$$

【例 27】 求解方程组(1) $\begin{cases}x+y=1\\x+y=2\end{cases}$,(2) $\begin{cases}x+y+z=1\\4x+2y+z=4\\3x+y=3\end{cases}$,观察解的情况。

```
> s:=solve({x+y=1,x+y=2});
```

$$s:=$$

solve()返回空集(NULL),表明方程组无解。

```
> solve({x+y+z=1,4*x+2*y+z=4,3*x+y=3});
```

$$\{y=-3x+3, z=-2x-2, x=x\}$$

方程组有无穷多组解，solve 返回方程组的通解，x 为自由变量。

【例 28】 求解非线性方程组(1) $\begin{cases}x^2+y=5\\x+y=3\end{cases}$，(2) $\begin{cases}x^2+y^2=1\\x-y=1\end{cases}$。

```
> solve({x^2+y=5,x+y=3});
```

$$\{y=1, x=2\}, \{x=-1, y=4\}$$

```
> solve({x^2+y^2=1,x-y=1});
```

$$\{y=0, x=1\}, \{y=-1, x=0\}$$

2.4.4 求解不等式

solve 函数可以用来求解不等式，其调用方式和求解方程相同。在不等式求解中，系统经常给出通过函数 RealRange 和 Open 表示的求解结果，其中 RealRange(a,b) 表示以 a,b 为端点的区间$[a,b]$，RealRange(Open(a),Open(b))表示不包括端点的开区间(a,b)。a,b 可以是数值、∞、$-\infty$。

【例 29】 求解不等式。

```
> solve((x-1)^2>9);  # 求解区域
```

$$\text{RealRange}(-\infty, \text{Open}(-2)), \text{RealRange}(\text{Open}(4), \infty)$$

```
> solve((x-1+a)*(x-2+a)*(x-3+a)<0,x);
```

$$\{x<1-a\}, \{2-a<x, x<3-a\}$$

```
> solve((x-2)*(x-3)*(x-4)<=0);
```

$$\text{RealRange}(-\infty, 2), \text{RealRange}(3, 4)$$

```
> solve({x+y<12,x^2=16});
```

$$\{y<16, x=-4\}, \{x=4, y<8\}$$

【例 30】 系数带符号的不等式。

```
> assume(a,real); solve(a*x<b,x);  # 设 a 为实数
```

$$\left\{\text{signum}(a\sim)x<\frac{\text{signum}(a\sim)b}{a\sim}\right\}$$

signum(x)是符号函数，当 $x>0$ 时值为 1，当 $x<0$ 时值为 -1，当 $x=0$ 时值为 0。

```
> assume(a>0); solve(a*x<b,x);  # 假设 a>0
```

$$\left\{x<\frac{b}{a\sim}\right\}$$

2.4.5 计算方程的数值解(fsolve)

fsolve(方程,变量,选项)

用数值方法求解方程的近似实数解,方程也可以是方程组或不等式(组),变量也可以是变量列表或集合,变量和选项都可以省略。fsolve 语句返回一个序列。无论方程的元素是准确数还是近似数,方程的解都按小数形式给出。精度由全局变量 Digits(默认值为 10)控制,用户可以改变 Digits 的值以提高求解的精度。如果方程是病态的,某些解可能会被遗漏。

【例 31】 用 fsolve 求解方程(组)。

```
> fsolve(tan(sin(x))=1);
                    0.9033391108
> fsolve(x^2-2*x+1);
                    1.000000000, 1.
> fsolve(x^5-3*x^3+1);
          -1.782308780, 0.7418139305, 1.668777593
```

为何 5 次多项式方程只有 3 个根?因为系统的默认值是实数域。在函数中加入 complex 选项,把求解的范围扩展到复数域,则有:

```
> fsolve(x^5-3*x^3+1,x,complex);
   -1.782308780, -0.3141413715-0.5954413283I, -0.3141413715
          +0.5954413283I, 0.7418139305, 1.668777593
```

求解例 26 中的方程组。

```
> fsolve({u+v+w=1,3*u+v=3,u-2*v-w=0}); # 给出数值解
     {u=0.80000000000,v=0.60000000000,w=-0.40000000000}
> Digits:=30; # 设置 30 位有效数字
                          30
> fsolve(sin(x),x,3..4); # 给出在区间[3,4]内的解
              3.14159265358979323846264338328
```

2.4.6 求解递归方程(rsolve)

rsolve(方程,函数,选项)

rsolve 求解满足递归方程的函数,用选项控制函数的表示形式。例如,选项 series表示解函数按级数形式输出,选项可省略。

【例 32】 求解递归方程。

```
> rsolve(f(n)=n*f(n-1),f);
```

$$\Gamma(n+1)f(0)$$

```
> rsolve({F(n)=F(n-1)+F(n-2),F(1..2)=1},F); # Fibonacci 数列
```

$$\frac{1}{5}\sqrt{5}\left(\frac{1}{2}+\frac{1}{2}\sqrt{5}\right)^n-\frac{1}{5}\sqrt{5}\left(\frac{1}{2}-\frac{1}{2}\sqrt{5}\right)^n$$

```
> rsolve(f(x)=diff(f(x),x),f,'series'); # 微分方程的形式幂级数解
```

$$f(x)=_c_1+x_c_1+\frac{1}{2}x^2_c_1+\frac{1}{6}x^3_c_1+\frac{1}{24}x^4_c_1+\frac{1}{120}x^5_c_1+O(x^6)$$

【例 33】 求解差分方程组 $\begin{cases} s_n = t_n - t_{n-1} \\ t_n = s_{n+1} - s_n \end{cases}$，初始条件是 $\begin{cases} s_0 = 0 \\ t_0 = 1 \end{cases}$。

```
> rsolve({s(n)=t(n)-t(n-1),t(n)=s(n+1)-s(n),s(0)=0,t(0)=1},
{s,t}, 'genfunc'(x));
```

$$\left\{s(x)=\frac{x}{1+x^2-3x},t(x)=-\frac{-1+x}{1+x^2-3x}\right\}$$

输出母函数 $s(x)=\sum_{n=0}^{\infty}s_nx^n$，$t(x)=\sum_{n=0}^{\infty}t_nx^n$。

2.5 三角函数及变换

2.5.1 三角函数及其反函数

表 2-3 列出了三角函数及其反函数。

表 2-3 三角函数及其反函数

sin(x)	cos(x)	tan(x)	csc(x)	sec(x)	cot(x)
正弦	余弦	正切	余割	正割	余切
arcsin(x)	arccos(x)	arctan(x)	arccsc(x)	arcsec(x)	arccot(x)
反正弦	反余弦	反正切	反余割	反正割	反余切

在默认情况下，系统会以弧度作为参数单位（π 弧度=180 角度），利用系统提供的基本函数，用户可以计算所有的三角函数。

【例 34】 计算三角函数值。

```
> sin(1.2);
```

0.9320390860

```
> sin(1.1+2.2I);
```

4.070953523+2.021725618I

```
> sin(arcsin(2)),cos(60 * Pi/180); # 化角度为弧度
```

$$2, \frac{1}{2}$$

利用 convert 函数也可以进行角度与弧度之间的互相转化。

```
> convert(Pi/2,degrees); # 化弧度为角度
```

90 degrees

```
> convert(60 * degrees, radians); # 化角度为弧度
```

$$\frac{1}{3}\pi$$

反三角函数同三角函数的用法类似，返回弧度值。反三角函数还可以用复合函数的形式表示。例如，(f@@3)(x)表示 $f(f(f(x)))$，(f@@(−3))(x)表示 $f^{-1}(f^{-1}(f^{-1}(x)))$。

```
> (sin@@(-1))(1/2);
```

$$\frac{1}{6}\pi$$

利用三角函数和反三角函数的组合，我们可以得到一些基本的三角恒等变化公式。

```
> sin(arccos(x));
```

$$\sqrt{1-x^2}$$

```
> cos(arctan(x));
```

$$\frac{1}{\sqrt{1+x^2}}$$

2.5.2 恒等变换与三角函数展开

三角函数的恒等变换是高中数学的基本内容。利用 combine 函数，用户可以完成和差化积、积化和差、半角倍角等一系列工作。

combine(表达式，变量，选项)

combine 可以将表达式中的多个和式、乘积、幂次项合并为一个和式、乘积、幂次项，将多个 Int、Sum、Limit 表达式合并为一个 Int、Sum、Limit 表达式。请从下面的

例题中理解“合并归一”。

【例 35】 观察 combine 和 expand 的作用效果。

```
> expand(sin(x+y));
```

$$\sin(x)\cos(y) + \cos(x)\sin(y)$$

```
> combine(sin(x) * cos(x)); # 化解乘号,即积化和差
```

$$\frac{1}{2}\sin(2x)$$

```
> combine(sin(x)+2sin(x)^2+3cos(x)^3);
```

$$\sin(x)+1-\cos(2x)+\frac{3}{4}\cos(3x)+\frac{9}{4}\cos(x)$$

```
> expand(cos(3 * x));
```

$$4\cos(x)^3-3\cos(x)$$

将两个积分式合并为一。

```
> combine(Int(sin(x)^2,x=a..b)-Int(cos(x)^2,x=a..b));
```

$$\int_a^b -\cos(2x)\mathrm{d}x$$

可以简单的理解为:combine 将三角函数变换为形如 $\sum_k a_k\cos(kx)+b_k\sin(kx)$ 的 Fourier 级数,而 expand 则将三角函数变换为形如 $\sum_{i,j} c_{i,j}\sin(x)^i\cos(x)^j$ 的幂级数。

利用 convert 函数,用户可以将三角函数化为只含有 sin、cos、tan 的形式。

【例 36】 三角函数之间、三角函数和指数间的转换。

```
> expr:=sec(x) * cot(x)-sin(x):
> convert(expr,tan);
```

$$\frac{1+\tan\left(\frac{1}{2}x\right)^2}{\left(1-\tan\left(\frac{1}{2}x\right)^2\right)\tan(x)}-\frac{2\tan\left(\frac{1}{2}x\right)}{1+\tan\left(\frac{1}{2}x\right)^2}$$

```
> exprb:=convert(expr,sincos);
```

$$\frac{1}{\sin(x)}-\sin(x)$$

用户还可以进行三角函数与指数以及反三角函数与对数间的互换。

```
> convert(tan(x),exp); # 三角函数和指数间的互换
```

$$-\frac{\mathrm{I}((\mathrm{e}^{(\mathrm{I}x)})^2-1)}{(\mathrm{e}^{(\mathrm{I}x)})^2+1}$$

```
> convert(exp(x^2)-2*exp(-x^2),cos);
```

$$\cos(-Ix^2)-I\cos\left(Ix^2+\frac{1}{2}\pi\right)-2\cos(Ix^2)+2I\cos\left(Ix^2+\frac{1}{2}\pi\right)$$

【例 37】 反三角函数和对数间的互换。

```
> convert(arctan(x),ln);
```

$$\frac{1}{2}I(\ln(1-Ix)-\ln(1+Ix))$$

2.5.3 双曲函数及其反函数

Maple 中也包含了双曲函数及其反函数(表 2-4),函数变量可为有理数、实数和复数。其中

$$\sinh(x)=\frac{e^x-e^{-x}}{2},\cosh(x)=\frac{e^x+e^{-x}}{2}$$

表 2-4 双曲函数及其反函数

sinh(x)	cosh(x)	tanh(x)	csch(x)	sech(x)	coth(x)
双曲正弦	双曲余弦	双曲正切	双曲余割	双曲正割	双曲余切
arcsinh(x)	arccosh(x)	arctanh(x)	arccsch(x)	arcsech(x)	arccoth(x)
反双曲正弦	反双曲余弦	反双曲正切	反双曲余割	反双曲正割	反双曲余切

【例 38】 双曲函数的计算和化简。

```
> sech(1+1.2*I);
```

$$0.3697083236-0.7242348129I$$

```
> convert(sinh(x),exp);
```

$$\frac{1}{2}e^x-\frac{1}{2}\frac{1}{e^x}$$

```
> simplify(sinh(x)^2-cosh(x)^2);
```

$$-1$$

```
> expand(cosh(x+y));
```

$$\cosh(x)\cosh(y)+\sinh(x)\sinh(y)$$

```
> simplify(convert(tanh(x),exp));
```

$$\frac{e^{(2x)}-1}{e^{(2x)}+1}$$

习　题

1. 计算下列多项式的值。

(1) $f(x)=x^3-2x-1, x=11,22,33$

(2) $g(x,y)=x^5y-x^4y^2+3x^3y-1, x=1, y=2; x=3, y=-2$

2. 已知 $f(x,y,z)=(x-y)^3-(2x-3y+z)^5$。

(1) 问 $f(x,y,z)$的展开式中有多少项?

(2) 将多项式 $f(x,y,z)$按 $x\to y\to z$ 降幂排序后,取出第一、第二和最后一项。

3. 分别在实数域和复数域因式分解下列各式。

(1) x^7-x^3

(2) x^4-y^4

(3) $81-x^4$

(4) $x^3+5x^2+11x+15$

(5) $a^3+b^3+c^3-3abc$

(6) $x^3y-1+xy+x^3+x-x^2y-y-x^2$

4. 对下列数列求和。

(1) $\sum_{k=0}^{100}(1+2^{-k})$

(2) $\sum_{k=1}^{\infty}\frac{1}{k(k+1)}$

(3) $\sum_{n=1}^{\infty}\frac{(n+1)^2}{n!}$

(4) $\sum_{i=1}^{30}\sum_{j=1}^{30}\frac{1}{i+j-1}$

5. 对下列数列求积。

(1) $\prod_{k=1}^{100}\frac{k}{k+2}$

(2) $\prod_{i=1}^{20}\prod_{j=1}^{20}\frac{i+j-1}{i+j+2}$

6. 用 solve 或移项函数 isolate 解下列方程。

(1) $3x^2+5(2x+1)=0$

(2) $(y-3)^2-(y+3)^2=9y(1-2y)$

(3) $abx^2+(a^4+b^4)x+a^3b^3=0, ab\neq 0$

(4) $x^2-(2m+1)x+m^2+m=0$

7. 解下列方程组,并用 eval 和 subs()函数验证方程组的解。

(1) $\begin{cases}\sqrt{3}x+\sqrt{3}y=\sqrt{7}\\ \sqrt{6}x-\sqrt{7}y=\sqrt{5}\end{cases}$

(2) $\begin{cases}u+v+w=1\\ 3u+v-w=3\\ u-2v-w=0\end{cases}$

(3) $\begin{cases}4x^2-9y^2=15\\ 2x-3y=5\end{cases}$

(4) $\begin{cases}x^2+2xy+y^2=9\\ (x-y)^2-3(x-y)=10\end{cases}$

8. 解下列不等式。

(1) $(x-2+a)(x-12+a)(x-112+a)<0$

(2) $\begin{cases}x+y<30\\ x^3>27\end{cases}$

(3) $\begin{cases}x^2<1\\ y^2\leqslant 1\\ x+y<\dfrac{1}{2}\end{cases}$

第3章　微　积　分

微积分是高等数学的基础，在实践中也有广泛的应用。本章主要介绍单变量和多变量微积分的基本运算，内容包括函数极限、导数、定积分、不定积分、数值积分、级数展开、Fourier 展开等。

在 Maple 中，微积分部分运算函数在内核中，可直接调用函数，如求导 diff、积分 int 等；部分函数在 student 函数包中，如换元 changevar、分部积分 intparts、线积分 Lineint、旋转体面积 SurfaceOfRevolution、旋转体体积 VolumeOfRevolution 等；Fourier 变换和 Laplace 变换在 inttrans 函数包中。

3.1　极限与连续

3.1.1　求极限(limit)

limit(f,x=a,dir)

计算$\lim\limits_{x\to a}f(x)$，其中 f 为代数表达式，x 为变量名，a 可为表达式、数值、∞或$-\infty$，dir 为极限逼近方向，可以取值 left(左极限)、right(右极限)、real(缺省值，实数轴的两方向的极限)或 complex(复平面上的所有方向的极限)。

【例 1】 计算单变量函数极限。

(1) 计算$\lim\limits_{x\to\infty}\dfrac{4x^3+2x^2+3}{x^3-1}$

```
> limit((4*x^3+2*x^2+3)/(x^3-1),x=infinity);
```

$$4$$

(2) 按定义计算 sin(x)的导数$\lim\limits_{h\to 0}\dfrac{\sin(x+h)-\sin x}{h}$

```
> limit((sin(x+h)-sin(x))/h,h=0);
```

$$\cos(x)$$

(3) 计算$\lim\limits_{x\to 0} x\sin\frac{1}{x}$(图 3-1)

```
> limit(x * sin(1/x),x=0);
```

$$0$$

```
> plot(x * sin(1/x),x=-0.5..0.5);
```

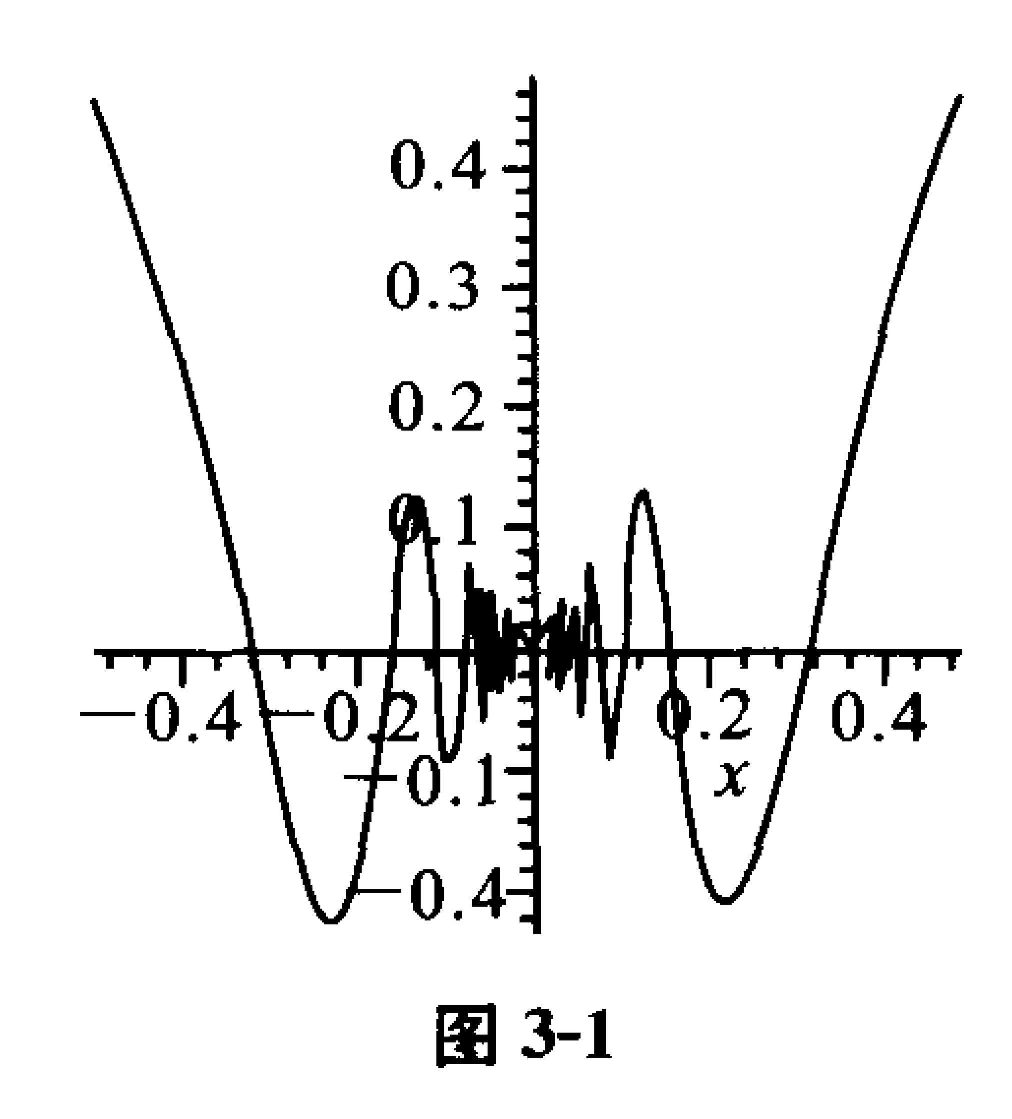


图 3-1

(4) 计算$\lim\limits_{x\to 0}\frac{1}{x}$

```
> limit(1/x, x=0); # 结果不同寻常
```

$$\text{undefined}$$

当函数极限不存在时,返回 undefined。从上面可看出,$1/x$ 在 $x=0$ 处的极限不存在,但是左右极限是存在的。

```
> limit(1/x,x=0,left),limit(1/x,x=0,right);
```

$$-\infty,\infty$$

(5) 计算$\lim\limits_{x\to 0}\sin\frac{1}{x}$

```
> limit(sin(1/x),x=0);
```

$$-1..1$$

函数 $\sin\frac{1}{x}$ 在 $x=0$ 处的极限不存在,输出"−1..1"表示函数值在区间[−1,1]中振荡。

(6) 计算$\lim\limits_{x\to\infty} x^a$

```
> limit(x^a,x=infinity);
```

$$\lim_{x\to\infty} x^a$$

当 Maple 无法计算函数的极限时,返回极限表达式。在上例中,因系统无法确定 a 的值,故无法计算。在增加 $a>0$ 的条件后,又可以计算了。

```
> limit(x^a,x=infinity) assuming a>0;
```

$$\infty$$

再看看与 limit 相对应的惰性函数 Limit 的作用。

(7) 计算$\lim\limits_{x\to 0}\dfrac{e^{ax}-e^{bx}}{\sin(ax)-\sin(bx)}$

> Limit((exp(a * x)－exp(b * x))/(sin(a * x)－sin(b * x)),x=0);

$$\lim_{x\to 0}\left(\frac{e^{(ax)}-e^{(bx)}}{\sin(ax)-\sin(bx)}\right)$$

> value(%); # 用 value 计算函数极限

$$1$$

(8) 计算$\lim\limits_{x\to\infty}\left(1+\dfrac{1}{x}\right)^x$

> Limit((1+1/x)^x,x=infinity)=limit((1+1/x)^x,x=infinity);

$$\lim_{x\to\infty}\left(1+\frac{1}{x}\right)^x=e$$

【例 2】 计算多变量函数极限。

(1) $\lim\limits_{\substack{x\to 1\\ y\to 1}}\left(5x^2y-\dfrac{4}{x+y^2}\right)$

> limit(5 * x^2 * y－4/(x+y^2),{x=1,y=1});

$$3$$

(2) $\lim\limits_{\substack{x\to 0\\ y\to 0}}\dfrac{x+y}{x-y}$(图 3-2)

> limits((x+y)/(x－y),{x=0,y=0}); # 极限不存在

undefined

> plot3d((x+y)/(x－y),x=－3..3,y=－3..3); # 看看函数图形

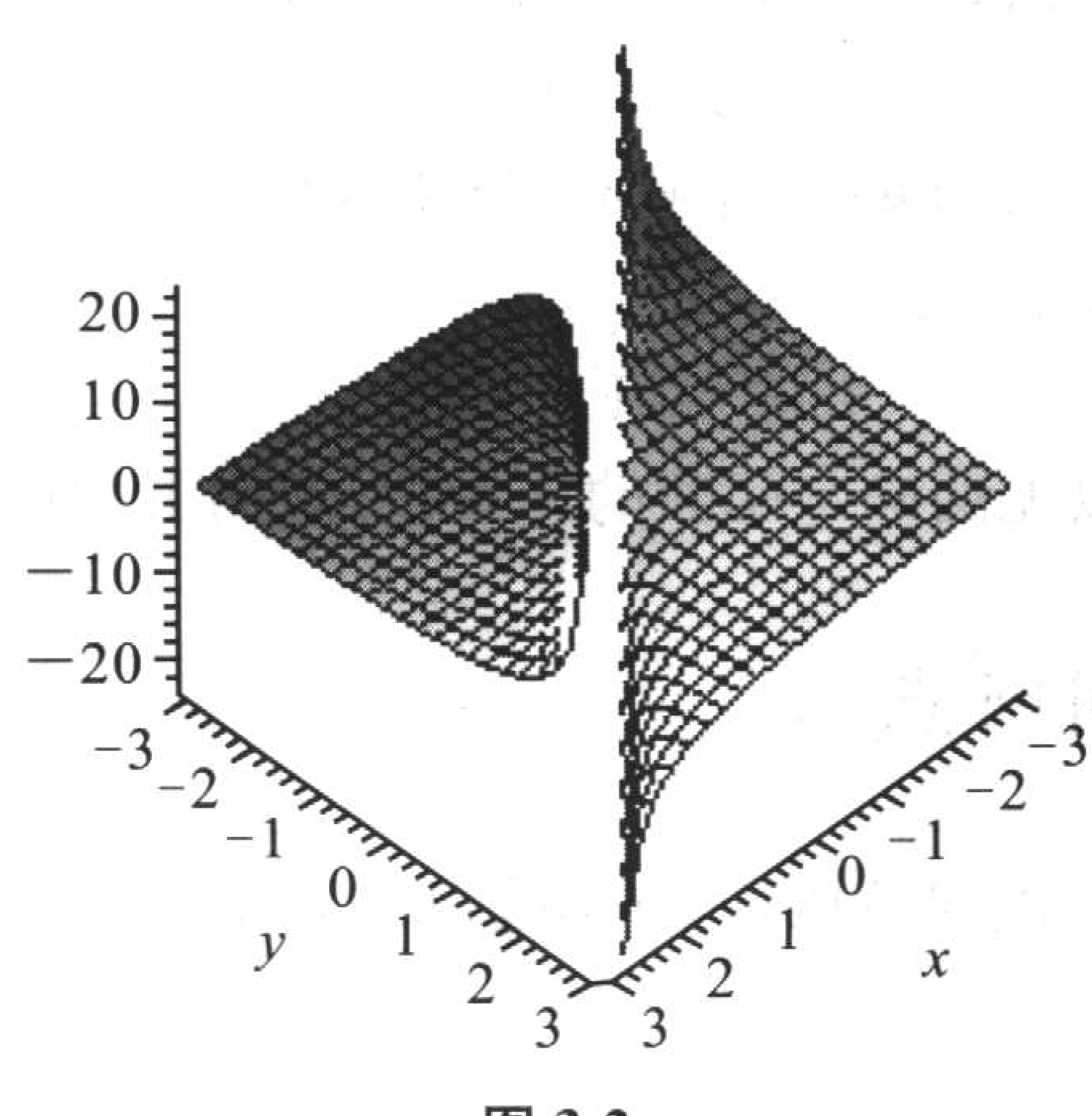


图 3-2

(3) $\lim\limits_{\substack{x\to 1\\ y\to 0}}\frac{\ln(x+\cos y)}{x^2+y^2}$

```
> limit(ln(x+cos(y))/(x^2+y^2),{x=1,y=0}); # 按兵不动,不做极限运算
```

$$\text{limit}\left(\frac{\ln(x+\cos(y))}{x^2+y^2},\{x=1,y=0\}\right)$$

```
> limit(limit(ln(x+cos(y))/(x^2+y^2),x=1),y=0);
```

$$\ln(2)$$

比较计算二重极限的两种方式,分别为(i) $\lim\limits_{\substack{x\to a\\ y\to b}}f(x,y)$和(ii) $\lim\limits_{y\to b}(\lim\limits_{x\to a}f(x,y))$。(ii)式将计算二元函数的二重极限化为两次计算一重极限,有时(i)式不能计算的极限(ii)式能胜任。

3.1.2 检验连续性

连续性是函数的一个重要性质,用 iscont 语句可以检验函数在给定区间上的连续性。

iscont(表达式,x=a..b,选项)

按选项检验表达式在区间 $a\sim b$ 上的连续性。当选项为 'open' 或缺省时,$a\sim b$ 是开区间;当选项为 'close' 时,$a\sim b$ 是闭区间,此时要求函数在端点的单边极限存在且有限。

【例 3】 检验函数的连续性。

```
> iscont(1/x,x=0..1); # 1/x 在开区间上连续
                              true
> iscont(1/x,x=0..1,'closed'); # 1/x 在闭区间上不连续
                              false
```

因为函数 $1/x$ 在 $x=0$ 处的左极限不是有限数,所以是不连续的。

```
> iscont(1/(x+a),x=0..1);
                              FAIL
```

因不知 a 的值,所以无法确定函数的连续性,返回 FAIL。

3.1.3 寻找间断点

寻找函数或表达式间断点的函数有 singular、discont 和 fdiscont。

singular(表达式,变量,区间)	表达式在区间内的奇点
discont(表达式,变量)	表达式的间断点
fdiscont(表达式,区间,分辨率,变量,选项)	数值方法求表达式在区间内的间断点

【例 4】 观察下列函数的间断点。

```
> singular(1/(x^2-3*x+2));
```

$$\{x=2\},\{x=1\}$$

```
> singular(tan(x),1..11);
```

$$\left\{x=\frac{1}{2}\pi\right\},\left\{x=\frac{7}{2}\pi\right\},\left\{x=\frac{3}{2}\pi\right\},\left\{x=\frac{5}{2}\pi\right\}$$

```
> discont(1/(x^2-3*x+2),x);
```

$$\{1,2\}$$

```
> singular(tan(x)),discont(tan(x),x);
```

$$\left\{x=_Z1\sim\pi+\frac{1}{2}\pi\right\},\left\{\pi_Z2\sim+\frac{1}{2}\pi\right\}$$

```
> discont(1/((x^3-1)*sin(x)),x);
```

$$\{1,\pi_Z1\sim\}$$

3.2 导数与微分

3.2.1 求导(diff)

```
diff(f,x1,…,xn)
diff(f,[x1,…,xn])
```

计算表达式 f 关于变量 $x_1,x_2,\cdots,x_n$ 的偏导数,其中 $x_1,x_2,\cdots,x_n$ 可重复出现。例如,diff(f,x1$m1,x2$m2,…,xn$mn)计算偏导数$\frac{\partial^{m_1+\cdots+m_n}f}{\partial x_1^{m_1}\cdots\partial x_n^{m_n}}$。在对多个变量求导时,依次调用 diff 求导,并且 diff 假定各偏导可以调换次序。

【例 5】 单变量求导。

(1) 计算 $\cos'(x)$, $\arctan'(x)$

```
> diff(cos(x),x),diff(arctan(x),x);
```

$$-\sin(x), \frac{1}{1+x^2}$$

(2) 计算$\frac{d}{dx}\left(\frac{f(x)}{g(x)}\right), \frac{d^4}{dx^4}\left(\frac{x}{\sin(x)}\right)$

```
> diff(f(x)/g(x), x);
```

$$\frac{\frac{d}{dx}f(x)}{g(x)}-\frac{f(x)\left(\frac{d}{dx}g(x)\right)}{g(x)^2}$$

```
> diff(x/sin(x), x$4);
```

$$-\frac{24\cos(x)^3}{\sin(x)^4}-\frac{20\cos(x)}{\sin(x)^2}+\frac{24x\cos(x)^4}{\sin(x)^5}+\frac{28x\cos(x)^2}{\sin(x)^3}+\frac{5x}{\sin(x)}$$

【例 6】 多变量求导。

(1) 计算$\frac{\partial^2 f(x,y)}{\partial x\partial y}$

```
> f: =x*y+x/y: diff(f,x,y);
```

$$1-\frac{1}{y^2}$$

```
> diff(sin(x*y),x,x); # 或用 diff(sin(x*y),x$2)
```

$$-\sin(yx)y^2$$

(2) 计算$\frac{\partial^5 \sin(xy)}{\partial x^2\partial y^3}, \frac{\partial^3 e^{xyz}}{\partial x\partial y\partial z}$

```
> diff(sin(x*y),y$3,x$2);
```

$$\cos(yx)y^2x^3+6\sin(yx)yx^2-6\cos(yx)x$$

```
> diff(exp(x*y*z),[x,y,z]);
```

$$e^{(xyz)}+3zyxe^{(xyz)}+y^2z^2x^2e^{(xyz)}$$

```
> diff(exp(x*y*z),[]); # 求导变量列表为空时,返回原函数
```

$$e^{(xyz)}$$

函数 Diff 是对应的 diff 的惰性函数,使用格式与 diff 相同。

```
> Diff(ln(x+sqrt(a^2+x^2)),x);
```

$$\frac{\partial}{\partial x}\ln(x+\sqrt{a^2+x^2})$$

```
> value(%);
```

$$\frac{1+\frac{x}{\sqrt{a^2+x^2}}}{x+\sqrt{a^2+x^2}}$$

```
> simplify(%);
```

$$\frac{1}{\sqrt{a^2+x^2}}$$

```
> Diff(exp(a * x),x $ 3)=diff(exp(a * x),x $ 3);
```

$$\frac{\partial^3}{\partial x^3}e^{(ax)}=a^3e^{(ax)}$$

```
> f:=(x,y)->ln(x+y^2): Diff(f(x,y),x $ 1,y $ 2)=diff(f(x,y),x $ 1,y $ 2);
```

$$\frac{\partial^3}{\partial y^2 \partial x}\ln(x+y^2)=\frac{8y^2}{(x+y^2)^3}-\frac{2}{(x+y^2)^2}$$

【例 7】 链式法则求导。

(1) 设 $u(t)=\ln(x^2+y^2+z^2),x=e^t,y=t,z=\ln t$,求 $u'(t)$

我们知道 $u'(t)=u_x x'(t)+u_y y'(t)+u_z z'(t)$。

```
> u:=log(x^2+y^2+z^2): x:=exp(t): y:=t: z:=log(t): diff(u,t);
```

$$\frac{2(e^t)^2+2t+\frac{2\ln(t)}{t}}{(e^t)^2+t^2+\ln(t)^2}$$

(2) 设 $u=f(x,xy,xyz)$,计算$\frac{\partial u}{\partial x}$

```
> unassign('f','x','y','z'); u:=f(x,x * y,x * y * z): diff(u,x);
```

$$(D_1(f))(x,xy,xyz)+(D_2(f))(x,xy,xyz)y+(D_3(f))(x,xy,xyz)yz$$

3.2.2 微分算子 D

函数 D 称为微分算子,它的功能与 diff 类似,但是只作用于函数形式,而不是函数值。

```
D(f)
D[i](f)
D[i](f)(x)
```

其中,f 为函数,i 为正整数或正整数序列,对应于被求导变量。D[i](f)计算关于第 i 个变量的偏导数。D[i](f)(x)计算点 x 处的导函数值。例如:

```
> D(sin)(a); # 或 D(x->sin(x))(a)
```

$$\cos(a)$$

D 作用于函数 sin,得到函数 cos,输入和输出都是函数名,不需给出自变量。

```
> D(x->x^2);
```

$$x\to 2x$$

【例 8】 微分算子 D 与 diff。

```
> (D(f))(x)-(diff(f(x),x));
```

$$(\mathrm{D}(f))(x)-\frac{\mathrm{d}}{\mathrm{d}x}f(x)$$

```
> simplify(%);
```

$$0$$

由上面的计算可以看到 D(f)(x)等效于 diff(f(x),x)。

```
> D(D(f)),(D@@n)(f);
```

$$(\mathrm{D}^{(2)})(f),(\mathrm{D}^{(n)})(f)$$

对 f 求高阶导数,可以逐次求导,也可以使用简洁的形式。

```
> f:=(x,y)->exp(x*y); D[1](f),D[2](f);
```

$$f=(x,y)\rightarrow \mathrm{e}^{(xy)}$$

$$(x,y)\rightarrow y\mathrm{e}^{(xy)},(x,y)\rightarrow x\mathrm{e}^{(xy)}$$

> D[1$3,2$2](f); # 计算$\dfrac{\partial^5 f(x,y)}{\partial^3 x\partial^2 y}$

$$(x,y)\rightarrow 6y\mathrm{e}^{(xy)}+6xy^2\mathrm{e}^{(xy)}+x^2y^3\mathrm{e}^{(xy)}$$

3.2.3 隐函数求导

implicitdiff(f,y,x)
implicitdiff(f,y,u,x)

从隐函数 $f(x,y)=0$ 计算(偏)导数$\dfrac{\partial y}{\partial x}$或$\dfrac{\partial u}{\partial x}$,其中 f 是关于 x 和 y 的表达式、方程或其集合,x 为自由变量或其序列,y 为相关变量或其集合,u 为 y 的元素或子集。

【例 9】 隐函数求导。

```
> f:=x^2+3*y^2=1:
> implicitdiff(f,y,x); # 计算 y'(x)
```

$$-\frac{1}{3}\frac{x}{y}$$

```
> implicitdiff(f,x,y); # 计算 x'(y)
```

$$-\frac{3y}{x}$$

```
> implicitdiff(f,y,x$2); # 计算 y''(x)
```

$$-\frac{1}{9}\frac{x^2+3y^2}{y^3}$$

【例 10】 已知理想气体的状态方程为 $PV=RT$,P 为压强,V 为体积,T 为温

度，R 为常数。证明$\frac{\partial V}{\partial T}\frac{\partial T}{\partial P}\frac{\partial P}{\partial V}=-1$。

```
> f：=P*V=R*T:
> implicitdiff(f,V,T)*implicitdiff(f,T,P)*implicitdiff(f,P,V) assuming
  R::constant;
```

$$-1$$

【例 11】 已知$\begin{cases}x+y+z=0\\ x\cdot y\cdot z=\sin(z)\end{cases}$，计算$\frac{\partial z}{\partial x}$，$\frac{\partial y}{\partial x}$。

```
> f：={x+y+z,x*y*z-sin(z)}: implicitdiff(f,{y,z},{y,z},x);
```

$$\left\{D(z)=\frac{z(x-y)}{-xz+xy-\cos(z)},D(y)=-\frac{-yz+xy-\cos(z)}{-xz+xy-\cos(z)}\right\}$$

```
> implicitdiff(f,{y,z},{y,z},x,notation=Diff);
```

$$\left\{\frac{\partial}{\partial x}z=\frac{z(x-y)}{-xz+xy-\cos(z)},\frac{\partial}{\partial x}y=-\frac{-yz+xy-\cos(z)}{-xz+xy-\cos(z)}\right\}$$

当方程的个数大于相关变量的个数的时候，或者导函数不存在的时候，implicitdiff 返回 FAIL。

```
> implicitdiff(f,z,x,y);
```

FAIL

3.3 积 分

3.3.1 单变量积分

int(f,x)	不定积分
int(f,x=a..b,选项)	定积分
Int(f,x)	不定积分的惰性形式
Int(f,x=a..b,选项)	定积分的惰性形式

其中选项有 continuous、CauchyPrincipalValue 和 AllSolutions。选项 continuous 不考虑积分中的不连续点，CauchyPrincipalValue 视积分在不连续点的左、右极限为同一极限(逼近速度相同)且正负无穷可以相互抵消，而 AllSolutions 则给出定积分在不同情况下所有的解。

【例 12】 计算单变量积分。

```
> int(sin(x),x);
```

$$-\cos(x)$$

```
> int(sin(x),x=0..Pi); # Pi 为数值(大写 P)
```

$$2$$

```
> int(sin(x),x=0..pi); # pi 为符号(小写 p)
```

$$1-\cos(\pi)$$

```
> Int(exp(-x^2),x)=int(exp(-x^2),x);
```

$$\int e^{(-x^2)}\,dx = \frac{1}{2}\sqrt{\pi}\,erf(x)$$

```
> plot(rhs(%),x=-10..10); # 画出积分函数
```

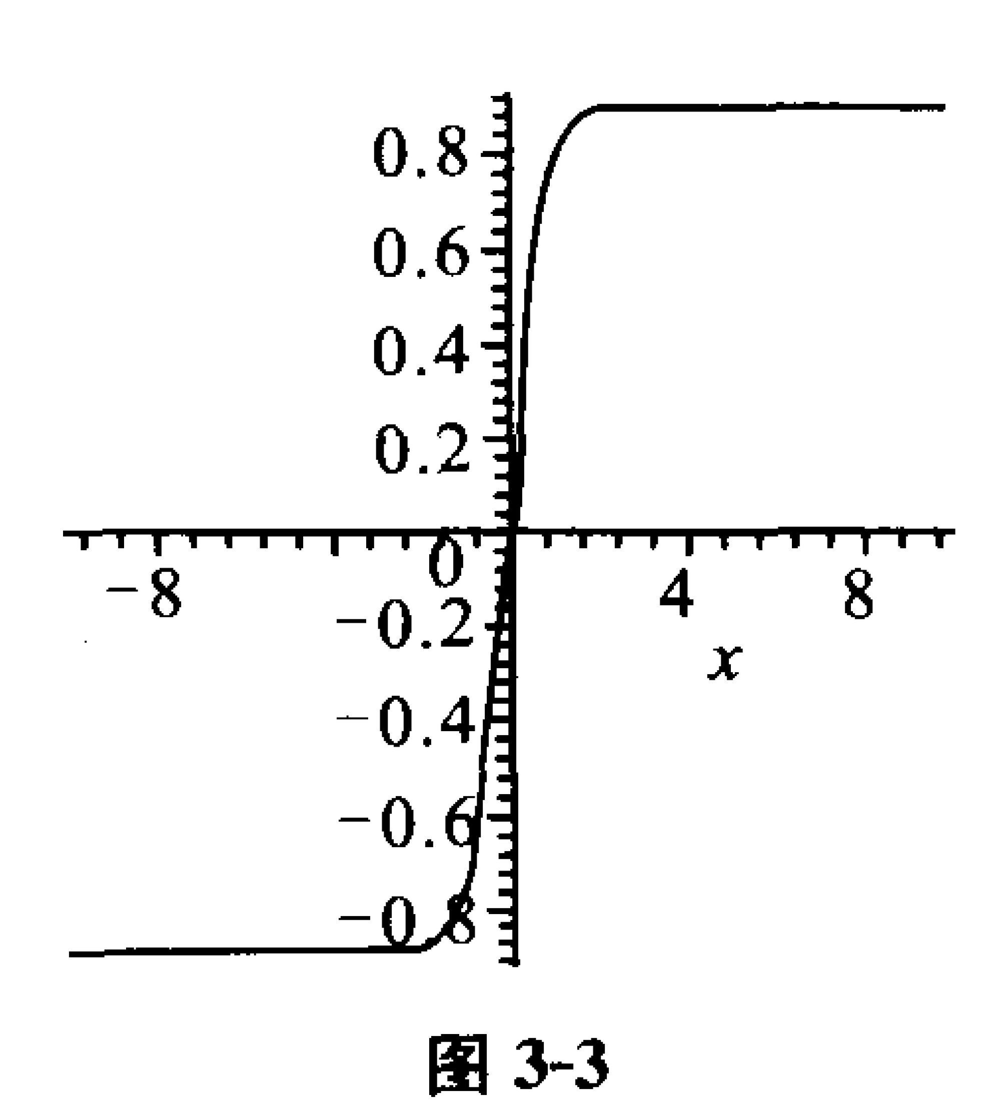


图 3-3

```
> f:=t^(a-1)*(1-t)^(b-1); Int(f,t=0..1)=int(f,t=0..1);
```

$$\int_0^1 t^{(a-1)}(1-t)^{(b-1)}\,dt=\frac{\Gamma(b)\Gamma(a)}{\Gamma(b+a)}$$

```
> int(1/x,x=-1..2); # 0 是 1/x 的奇点
```

undefined

```
> int(1/x,x=-1..2,'CauchyPrincipalValue');
```

$$\ln(2)$$

```
> int(1/x,x=a..2); # 为何没有计算结果?
```

Warning, unable to determine if 0 is between a and 2; try to use assumptions or set _EnvAllSolutions to true

$$\int_a^2 \frac{1}{x}\,dx$$

```
> int(1/x,x=a..2,'AllSolutions');
```

$$\begin{cases} \text{undefined} & a<0 \\ \infty & a=0 \\ -\ln(a)+\ln(2) & 0<a \end{cases}$$

```
> int(1/(x+a),x=0..2);
```

$$\begin{cases} \text{undefined} & \text{And}(a<0,-2<a) \\ -\ln(a)+\ln(a+2) & \text{otherwise} \end{cases}$$

```
> int(1/(x+a),x=0..2,'continuous');
```

$$-\ln(a)+\ln(a+2)$$

3.3.2 重　积　分

在 Maple 中，重积分是通过多次调用单变量来实现的。例如：

int(int(f(x,y),x),y)	$\iint f(x,y)\mathrm{d}x\mathrm{d}y$
int(int(f(x,y),x=a..b),y=c..d)	$\int_c^d\int_a^b f(x,y)\mathrm{d}x\mathrm{d}y$
int(int(int(f(x,y,z),x),y),z)	$\iiint f(x,y,z)\mathrm{d}x\mathrm{d}y\mathrm{d}z$
int(int(int(g(x,y,z), x=a..b),y=c..d),z=e..f)	$\int_e^f\int_c^d\int_a^b g(x,y,z)\mathrm{d}x\mathrm{d}y\mathrm{d}z$

而 student 函数包则提供了计算二重积分和三重积分的惰性函数 Doubleint 和 Tripleint。它们的用法如下：

```
Doubleint(f,x,y)
Doubleint(f,x,y,Domain)
Doubleint(f,x=a..b,y=c..d)
Tripleint(f,x,y,z)
Tripleint(f,x,y,z,Domain)
Tripleint(g,x=a..b,y=c..d,z=e..f)
```

其中 a,b,c,d,e,f 为积分区域端点，Domain 为积分区域的名称。

【例 13】 计算重积分 $\iint\limits_{-1\leqslant x,y\leqslant 1} \mathrm{e}^{x+y}\mathrm{d}x\mathrm{d}y$，$\iint\limits_{0\leqslant x<y\leqslant 1} 2\sqrt{x^2+y^2}\mathrm{d}x\mathrm{d}y$，$\iiint\limits_{\substack{x+y+z\leqslant 1\\ 0\leqslant x,y,z}} \frac{\mathrm{d}x\mathrm{d}y\mathrm{d}z}{(1+x+y+z)^3}$。

```
> int(int(exp(x+y),x=-1..1),y=-1..1);
```

$$\mathrm{e}^{(-2)}-2+\mathrm{e}^2$$

```
> with(student);
```

[D, Diff, Doubleint, Int, Limit, Lineint, Product, Sum, Tripleint, changevar, completesquare, distance, equate, integrand, intercept, intparts,

leftbox, leftsum, makeproc, middlebox, middlesum, midpoint, powsubs, rightbox, rightsum, showtangent, simpson, slope, summand, trapezoid]

```
> Doubleint(exp(x+y),x=-1..1,y=-1..1);
```

$$\int_{-1}^{1}\int_{-1}^{1} e^{(x+y)}\,dx\,dy$$

惰性函数 Doubleint 无法直接计算出数值积分的结果,利用 value 计算其值。

```
> value(%);
```

$$e^{(-2)}-2+e^{2}$$

```
> Doubleint(2*sqrt(x^2+y^2),x=0..y,y=0..1);
```

$$\int_{0}^{1}\int_{0}^{y} 2\sqrt{x^2+y^2}\,dx\,dy$$

```
> value(%); # 上式中 x=0..y 与 y=0..1 次序不可交换
```

$$\frac{1}{3}\sqrt{2}+\frac{1}{3}\ln(1+\sqrt{2})$$

```
> Doubleint(2*sqrt(x^2+y^2),x=0..y,y=0..1) =
  int(int(2*sqrt(x^2+y^2),x=0..y),y=0..1);
```

$$\int_{0}^{1}\int_{0}^{y} 2\sqrt{x^2+y^2}\,dx\,dy = \frac{1}{3}\sqrt{2}+\frac{1}{3}\ln(1+\sqrt{2})$$

```
> int(int(int(1/(1+x+y+z)^3,x=0..1-y-z),y=0..1-z),z=0..1);
```

$$-\frac{5}{16}+\frac{1}{2}\ln(2)$$

```
> Tripleint(1/(1+x+y+z)^3,x=0..1-y-z,y=0..1-z,z=0..1);
```

$$\int_{0}^{1}\int_{0}^{1-z}\int_{0}^{1-y-z} \frac{1}{(1+x+y+z)^3}\,dx\,dy\,dz$$

```
> value(%);
```

$$-\frac{5}{16}+\frac{1}{2}\ln(2)$$

3.3.3 换 元 积 分

为了帮助学生在学习积分过程中掌握基本的方法,Maple 提供了换元积分与分部积分函数。

```
changevar(s,f)
changevar(s,f,u)
```

其中 f 是积分表达式(假设积分变量名为 x),s 为形如 $h(x)=g(u)$ 的表达式,u 是新的积分变量。当 f 为重积分的时候,u 为新变量列表。使用 changevar 之前,需要先调用 student 函数包。changevar 也可以用于极限、求和表达式的变量替换。例如:

```
> with(student): A := Int(f(x),x);
```

$$A := \int f(x)\mathrm{d}x$$

```
> changevar(x=g(t),A,t);
```

$$\int f(g(t))\left(\frac{\mathrm{d}}{\mathrm{d}t}g(t)\right)\mathrm{d}t$$

```
> B := Int(sqrt(1-x^2),x=0..1): B=changevar(x=sin(u),B,u);
```

$$\int_0^1 \sqrt{1-x^2}\,\mathrm{d}x = \int_0^{\frac{1}{2}\pi} \sqrt{1-\sin(u)^2}\cos(u)\mathrm{d}t$$

```
> value(%);
```

$$\frac{1}{4}\pi=\frac{1}{4}\pi$$

【例 14】 用换元函数计算积分。

(1) $\int \sqrt{2+x-x^2}\,\mathrm{d}x$ 。

分析：因为 $\sqrt{2+x-x^2}=\sqrt{\frac{9}{4}-\left(x-\frac{1}{2}\right)^2}$，令 $x-\frac{1}{2}=\frac{3}{2}\sin t$

```
> changevar(x-1/2=3*sin(t)/2,Int(sqrt(2+x-x^2),x),t);
```

$$\int \frac{3}{2}\sqrt{\frac{5}{2}+\frac{3}{2}\sin(t)-\frac{1}{4}(1+3\sin(t))^2}\cos(t)\mathrm{d}t$$

```
> value(%);
```

$$\frac{9}{8}\sqrt{1-\sin(t)^2}\sin(t)+\frac{9}{8}t$$

```
> simplify(subs(t=arcsin((2*x-1)/3),%)); # 将变量 t 还原为 x
```

$$\frac{1}{2}\sqrt{2+x-x^2}\,x-\frac{1}{4}\sqrt{2+x-x^2}+\frac{9}{8}\arcsin\left(\frac{2}{3}x-\frac{1}{3}\right)$$

```
> int(sqrt(2+x-x^2),x);
```

$$-\frac{1}{4}(1-2x)\sqrt{2+x-x^2}+\frac{9}{8}\arcsin\left(\frac{2}{3}x-\frac{1}{3}\right)$$

(2) $\int \sqrt{a^2-x^2}\,\mathrm{d}x, a>0$ 。

令 $x=a\sin t$

```
> changevar(x=a*sin(t),Int(sqrt(a^2-x^2),x),t);
```

$$\int \sqrt{a^2-a^2\sin(t)^2}\,a\cos(t)\mathrm{d}t$$

```
> value(%) assuming a>0;
```

$$a\left(\frac{1}{2}\sin(t)\sqrt{a^2-a^2\sin(t)^2}+\frac{1}{2}a\arcsin(\sin(t))\right)$$

```
> subs(sin(t)=x/a,%); # 将变量 t 还原为 x
```

$$a\left(\frac{1}{2}\frac{x\sqrt{a^2-x^2}}{a}+\frac{1}{2}a\arcsin\left(\frac{x}{a}\right)\right)$$

```
> int(sqrt(a^2-x^2),x) assuming a>0;
```

$$\frac{1}{2}x\sqrt{a^2-x^2}+\frac{1}{2}a^2\arcsin\left(\frac{x}{a}\right)$$

【例 15】 重积分的变量替换。

```
> S:=Doubleint(exp(-x^2-y^2),x,y);
```

$$S:=\iint e^{(-x^2-y^2)}\,dx\,dy$$

```
> changevar({x=r*cos(t),y=r*sin(t)},S,[r,t]);
```

$$\iint \frac{|r|}{e^{(r^2)}}\,dr\,dt$$

3.3.4 分部积分

intparts(f,u)

计算 $uv-\int v\mathrm{d}u$，假设惰性积分 f 可写为 $\int u\mathrm{d}v$ 的形式。intparts 是惰性函数，它的运算结果中仍有积分式，还要调用 value 等函数才能算出积分值。使用 intparts 之前，需要先调用 student 函数包。

【例 16】 计算 $\int \ln x\mathrm{d}x$，$\int x\cos x\mathrm{d}x$，$\int \frac{x^2}{\sqrt{x^2+a^2}}\mathrm{d}x$，$\int x^2 e^x\mathrm{d}x$。

```
> with(student):
> intparts(Int(ln(x),x),ln(x));
```

$$\ln(x)x-\int 1\,dx$$

```
> value(%);
```

$$\ln(x)\,x-x$$

```
> intparts(Int(x*cos(x),x),x);
```

$$\sin(x)x-\int \sin(x)\,dx$$

```
> value(%);
```

$$\cos(x)+\sin(x)\,x$$

```
> intparts(Int(x^2/sqrt(x^2+a^2),x),x);
```

$$x\sqrt{x^2+a^2}-\int\sqrt{x^2+a^2}\,\mathrm{d}x$$

> value(%)；

$$\frac{1}{2}x\sqrt{x^2+a^2}-\frac{1}{2}a^2\ln(x+\sqrt{x^2+a^2})$$

> intparts(Int(x^2 * exp(x),x),x^2)；# 对 x^2 做第1次分部积分

$$x^2\mathrm{e}^x-\int 2x\mathrm{e}^x\,\mathrm{d}x$$

> intparts(%,x)；# 对 x 做第2次分部积分

$$x^2\mathrm{e}^x-2x\mathrm{e}^x+\int 2\mathrm{e}^x\,\mathrm{d}x$$

> value(%)；# 计算积分值

$$x^2\mathrm{e}^x-2x\mathrm{e}^x+2\mathrm{e}^x$$

3.3.5 曲线积分

student 函数包中的 Lineint 函数可以计算曲线积分。

```
Lineint(f,x,y)
Lineint(f,x,y,z)
```

【例17】 曲线积分计算公式。

> with(student)：

> Lineint(f(x,y),x,y)；

$$\int f(x(y),y)\sqrt{\left(\frac{\mathrm{d}}{\mathrm{d}y}x(y)\right)^2}\,\mathrm{d}y$$

> Lineint(f(x,y),x,y=a..b)；

$$\int_a^b f(x(y),y)\sqrt{\left(\frac{\mathrm{d}}{\mathrm{d}y}x(y)\right)^2}\,\mathrm{d}y$$

> Lineint(f(x,y),x,y,t)；

$$\int f(x(t),y(t))\sqrt{\left(\frac{\mathrm{d}}{\mathrm{d}t}x(t)\right)^2+\left(\frac{\mathrm{d}}{\mathrm{d}t}y(t)\right)^2}\,\mathrm{d}t$$

> Lineint(f(x,y,z),x=u,y=v,z)；

$$\int f(u,v,z)\sqrt{\left(\frac{\partial}{\partial z}u\right)^2+\left(\frac{\partial}{\partial z}v\right)^2}\,\mathrm{d}z$$

Lineint(f(x,y),x,y)计算 $f(x,y)$关于参数 y 的线积分，y 的范围可以指定 y=a..b。Lineint 只做简单的化简，欲做进一步的计算可以使用 value 函数。

【例 18】 (1)计算摆线 $x=a(t-\sin t)$, $y=a(1-\cos t)$, $0\leqslant t\leqslant 2\pi$ 的长度, $a>0$。

```
> Lineint(1,x=a*(t-sin(t)),y=a*(1-cos(t)),t=0..2*Pi);
```

$$\int_0^{2\pi}\sqrt{\left(\frac{\partial}{\partial t}(a(t-\sin(t)))\right)^2+\left(\frac{\partial}{\partial t}(a(1-\cos(t)))\right)^2}\,dt$$

```
> value(%) assuming a>0;
```

$$8a$$

(2) 计算曲线积分 $\int_L (x^2+y^2+z^2)\mathrm{d}l$,其中 L 是螺旋线 $x=R\cos t$, $y=R\sin t$, $z=kt$ 在 $0\leqslant t\leqslant 2\pi$ 的弧段。

```
> Lineint(x^2+y^2+z^2,x=R*cos(t),y=R*sin(t),z=k*t,t=0..2*Pi);
```

$$\int_0^{2\pi}(R^2\cos(t)^2+R^2\sin(t)^2+k^2t^2)\sqrt{\left(\frac{\partial}{\partial t}(R\cos(t))\right)^2+\left(\frac{\partial}{\partial t}(R\sin(t))\right)^2+\left(\frac{\partial}{\partial t}(kt)\right)^2}\,dt$$

```
> value(%);
```

$$2\sqrt{k^2+R^2}R^2\pi+\frac{8}{3}\sqrt{k^2+R^2}k^2\pi^3$$

3.3.6 旋转曲面积分

SurfaceOfRevolution(f(x),x=a..b)

计算平面曲线 $y=f(x)$, $x\in[a,b]$绕 x 轴旋转所得的旋转曲面的表面积。而

VolumeOfRevolution(f(x),x=a..b)

则是计算平面曲线 $y=f(x)$, $x\in[a,b]$绕 x 轴旋转所围成的旋转体的体积。用户还可以通过设置选项来改变旋转的方式。

SurfaceOfRevolution(f(x),x=a..b,选项)
VolumeOfRevolution(f(x),x=a..b,选项)

在使用 SurfaceOfRevolution 和 VolumeOfRevolution 函数之前,需要先调用 student[Calculus1]子函数包。

【例 19】 请观察 SurfaceOfRevolution 和 VolumeOfRevolution 函数中选项的效果。

```
> with(Student[Calculus1]);
```

[AntiderivativePlot, AntiderivativeTutor, ApproximateInt, ApproximateIntTutor, ArcLength, ArcLengthTutor, Asymptotes, Clear, CriticalPoints, CurveAnalysisTutor, DerivativePlot, DerivativeTutor, DiffTutor, ExtremePoints,

FunctionAverage, FunctionAverageTutor, FunctionChart, FunctionPlot, GetMessage, GetNumProblems, GetProblem, Hint, InflectionPoints, IntTutor, Integrand, InversePlot, InverseTutor, LimitTutor, MeanValueTheorem, MeanValueTheoremTutor, NewtonQuotient, NewtonsMethod, NewtonsMethodTutor, PointInterpolation, RiemannSum, RollesTheorem, Roots, Rule, Show, ShowIncomplete, ShowSteps, Summand, SurfaceOfRevolution, SurfaceOfRevolutionTutor, Tangent, TangentSecantTutor, TangentTutor, TaylorApproximation, TaylorApproximationTutor, Understand, Undo, VolumeOfRevolution, VolumeOfRevolutionTutor, WhatProblem]

```
> SurfaceOfRevolution(f(x),x=a..b);
```

$$\int_a^b 2\pi\,|\,f(x)\,|\sqrt{\left(\frac{\mathrm{d}}{\mathrm{d}x}f(x)\right)^2+1}\,\mathrm{d}x$$

```
> SurfaceOfRevolution(sin(x),0..2*Pi);
```

$$4\pi\sqrt{2}+4\pi\ln(1+\sqrt{2})$$

```
> SurfaceOfRevolution(sin(x),0..2*Pi,output=plot);
```

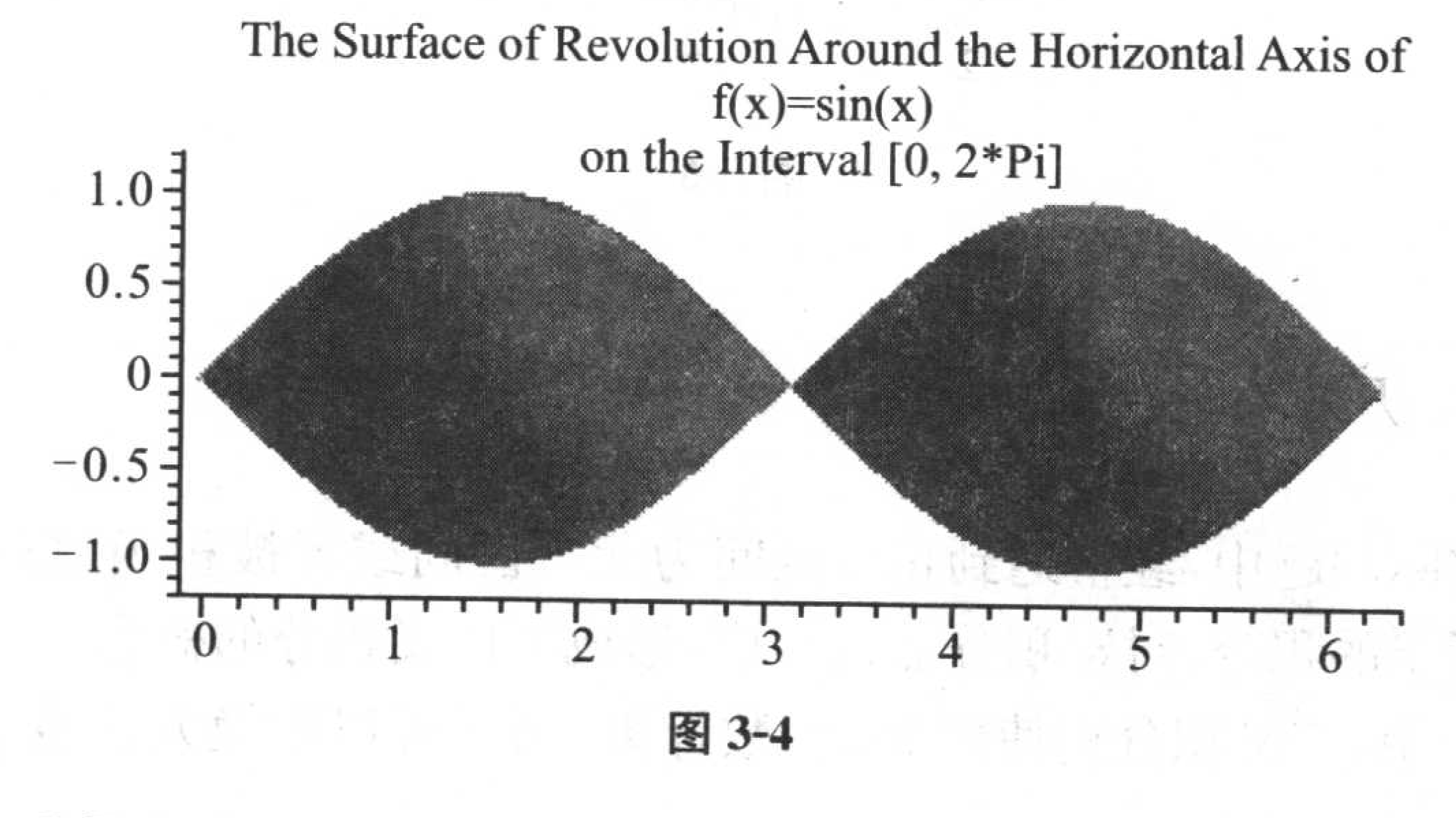


图 3-4

```
> SurfaceOfRevolution(sin(x),0..2*Pi,output=plot,axis=vertical);
```

The Surface of Revolution Around the Vertical Axis of
f(x)=sin(x)
on the Interval [0, 2*Pi]

图 3-5

```
> VolumeOfRevolution(f(x),x=a..b);
```

$$\int_a^b \pi f(x)^2 \mathrm{d}x$$

```
> VolumeOfRevolution(sin(x)+1,x=0..2*Pi);
```

$$3\pi^2$$

```
> VolumeOfRevolution(sin(x)+1,x=0..2*Pi,output=plot);
```

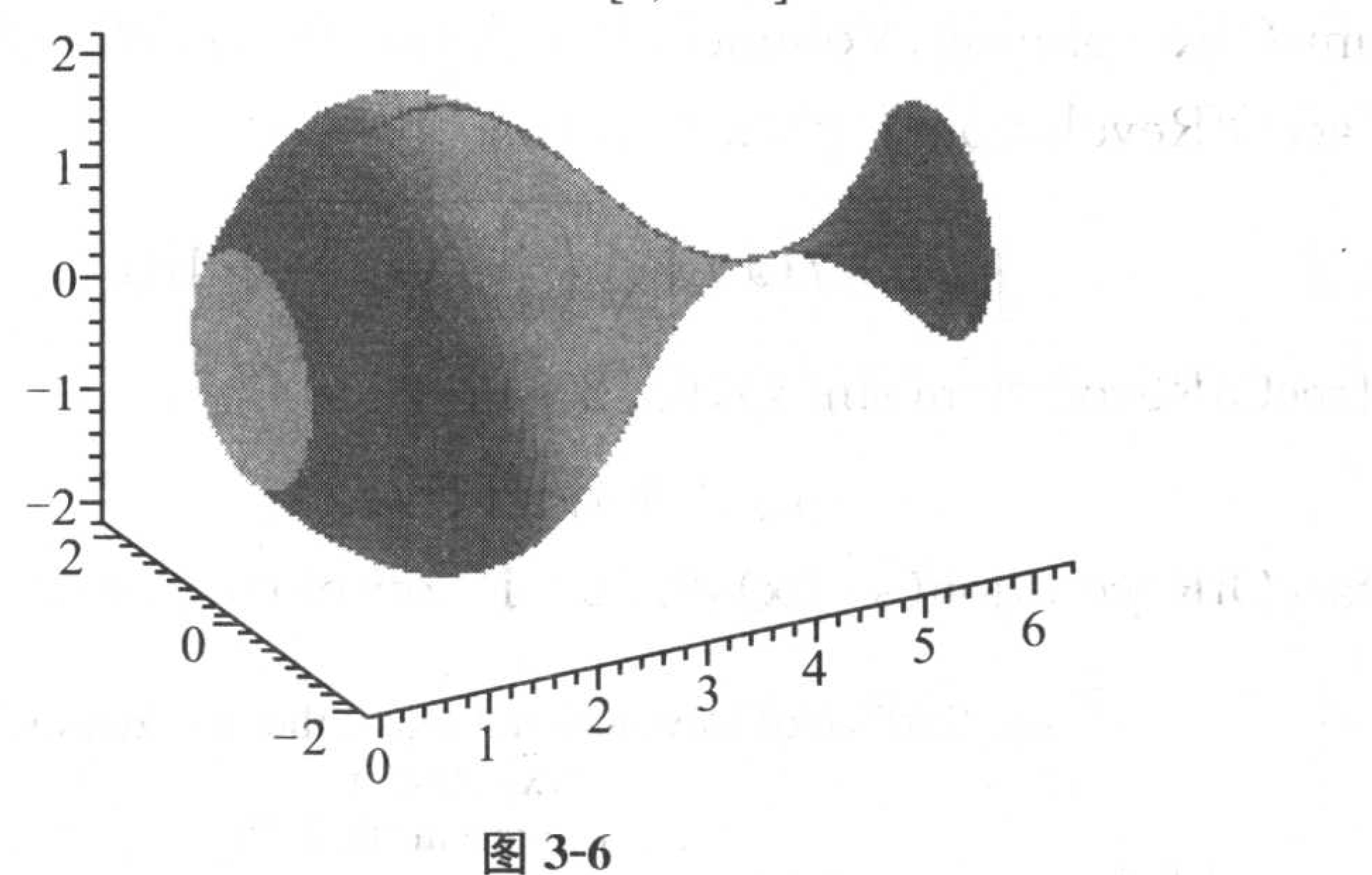


图 3-6

3.3.7 广义积分

在一些实际问题中，经常遇到积分区间为无穷区间或者被积函数在积分区间上具有无穷间断点的积分，对定积分加以推广，形成了广义积分的概念。

在 Maple 中，广义积分仍使用 int 函数计算，同样可以用换元函数 changevar 和分部积分函数 intparts。

【例 20】 计算积分端点为无穷的广义积分 $\int_{-\infty}^{+\infty} \frac{\mathrm{d}x}{1+x^2}, \int_{\mathrm{e}}^{+\infty} \frac{\mathrm{d}x}{x(\ln x)^2}$。

```
> Int(1/(1+x^2),x=-infinity..infinity) =
  int(1/(1+x^2),x=-infinity..infinity);
```

$$\int_{-\infty}^{\infty} \frac{1}{1+x^2} \mathrm{d}x = \pi$$

```
> Int(1/(x*log(x)^2),x=exp(1)..infinity) =
  int(1/(x*log(x)^2),x=exp(1)..infinity);
```

$$\int_{\mathrm{e}}^{\infty} \frac{1}{x\ln(x)^2} \mathrm{d}x = 1$$

【例 21】 计算积分区间含无穷间断点的瑕积分 $\int_0^1 \frac{\arcsin x}{\sqrt{1-x^2}}dx$ 。

```
> Int(arcsin(x)/sqrt(1-x^2),x=0..1) =
  int(arcsin(x)/sqrt(1-x^2),x=0..1);
```

$$\int_0^1 \frac{\arcsin x}{\sqrt{1-x^2}}dx = \frac{1}{8}\pi^2$$

3.4 数值积分

在数值计算方法中有许多数值积分的近似算法，如矩形公式、梯形公式、辛普森公式等。Maple 的 student 函数包中也有与这些算法相对应的函数，包括左矩形公式(leftsum)、中矩形公式(middlesum)、右矩形公式(rightsum)、梯形公式(trapezoid)、辛普森公式(simpson)等。

```
leftsum(f(x),x=a..b,n)
leftbox(f(x),x=a..b,n,绘图选项)
middlesum(f(x),x=a..b,n)
middlebox(f(x),x=a..b,n,绘图选项)
rightsum(f(x),x=a..b,n)
rightbox(f(x),x=a..b,n,绘图选项)
trapezoid(f(x),x=a..b,n)
simpson(f(x),x=a..b,n)
```

其中 $f(x)$ 为被积函数，a,b 为积分区间，n 为区间分段数(缺省值 4)。例如：

```
> with(student):
> leftbox(sin(x) * ln(x),x=1..4,6,color=red);
```

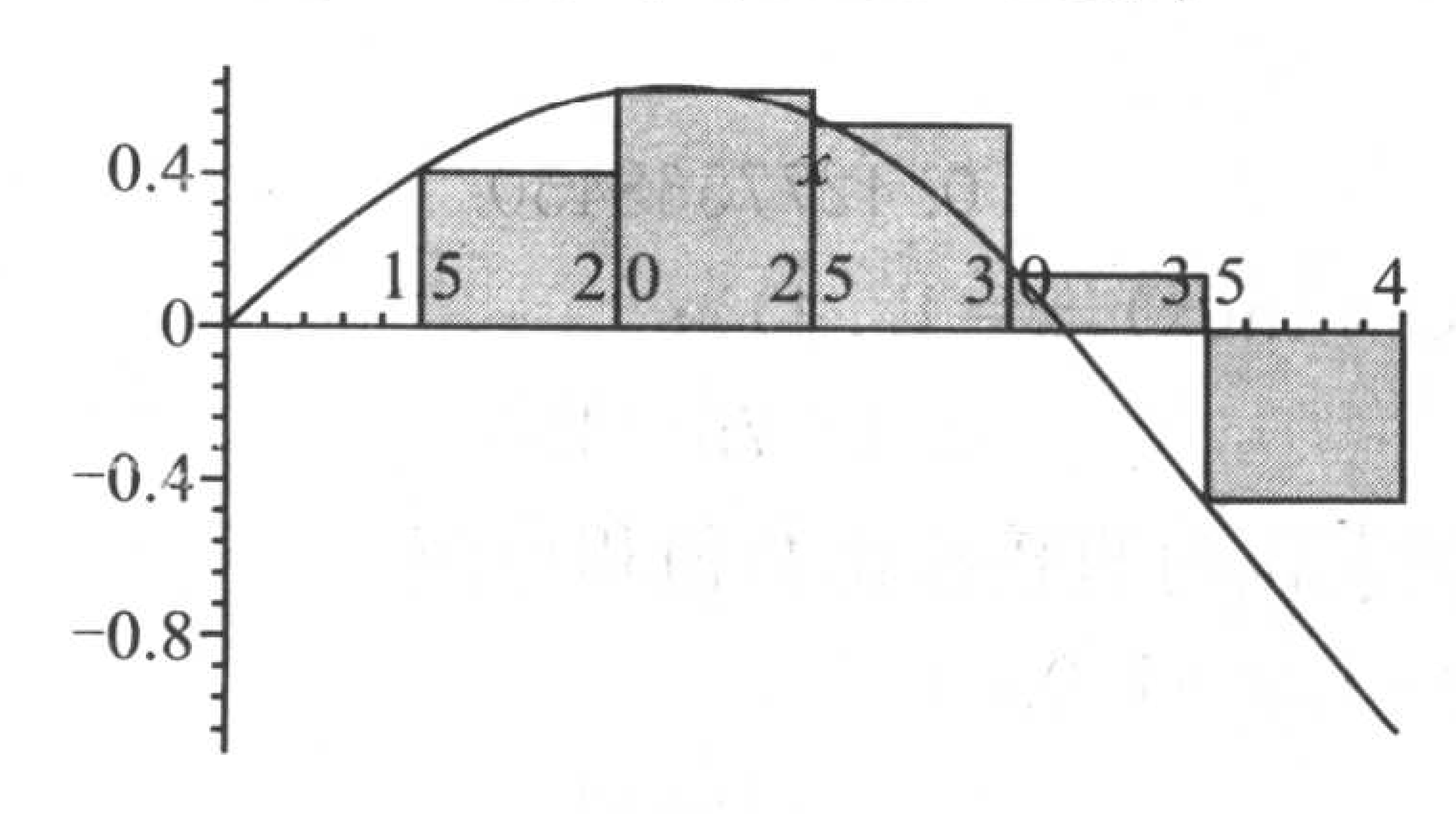


图 3-7

```
> leftsum(sin(x) * ln(x),x=1..4,6);
```

$$\frac{1}{2}\left(\sum_{i=0}^{5}\sin\left(1+\frac{1}{2}i\right)\ln\left(1+\frac{1}{2}i\right)\right)$$

```
> value(%);
```

$$\frac{1}{2}\sin\left(\frac{3}{2}\right)\ln\left(\frac{3}{2}\right)+\frac{1}{2}\sin(2)\ln(2)+\frac{1}{2}\sin\left(\frac{5}{2}\right)\ln\left(\frac{5}{2}\right)$$
$$+\frac{1}{2}\sin(3)\ln(3)+\frac{1}{2}\sin\left(\frac{7}{2}\right)\ln\left(\frac{7}{2}\right)$$

```
> evalf(%);
```

$$0.6493443882$$

```
> simpson(sin(x) * ln(x),x=1..4);
```

$$\frac{1}{2}\sin(4)\ln(2)+\sum_{i=1}^{2}\sin\left(\frac{1}{4}+\frac{3}{2}i\right)\ln\left(\frac{1}{4}+\frac{3}{2}i\right)$$
$$+\frac{1}{2}\left(\sum_{i=1}^{1}\sin\left(1+\frac{3}{2}i\right)\ln\left(1+\frac{3}{2}i\right)\right)$$

```
> simpson(sin(x) * ln(x),x=1..4,6);
```

$$\frac{1}{3}\sin(4)\ln(2)+\frac{2}{3}\left(\sum_{i=1}^{3}\sin\left(\frac{1}{2}+i\right)\ln\left(\frac{1}{2}+i\right)\right)$$
$$+\frac{1}{3}\left(\sum_{i=1}^{2}\sin(1+i)\ln(1+i)\right)$$

```
> evalf(%);
```

$$0.4291629716$$

计算定积分的数值近似时，更为一般的方法是使用 evalf 函数，或者用浮点数的形式表示积分的范围。

evalf(表达式)

最常用的是 evalf(Int(…))形式，这样可以避免调用 int 函数。

```
> int(sin(x) * ln(x),x=1..4);
```

$$-Ci(1)-2\cos(4)\ln(2)+Ci(4)$$

```
> evalf(%);
```

$$0.4277568450$$

```
> evalf(Int(sin(x) * ln(x),x=1..4));
```

$$0.4277568450$$

用浮点表示积分的范围时直接给出数值积分值。

```
> int(sin(x) * ln(x),x=1.0..4.0);
```

$$0.4277568450$$

下面我们比较以上 3 种方法的运行时间。

```
> restart; time(int(sin(x) * ln(x),x=1.0..4.0));
  time(evalf(Int(sin(x) * ln(x),x=1..4)));
  time(evalf(int(sin(x) * ln(x),x=1..4)));
```

$$0.015$$

$$0.016$$

$$0.218$$

由此可见,evalf(int(…))所花费的时间要远大于 evalf(Int(…))所花费的时间,而 evalf(int(浮点))与 evalf(Int(…))花费的时间几乎一样。

3.5 级 数

3.5.1 幂级数展开(series)

series(f(x),x=a,n)

给出 $f(x)$在 $x=a$ 处的 n 次 Taylor 展式或 n 项 Laurent 展式。如果只写 x,表示在 $x=0$ 处展开。n 为非负整数,缺省值是 6。

【例 22】 幂级数展开。

```
> series(f(x),x);
```

$$f(0)+(\mathrm{D}(f))(0)x+\frac{1}{2}((\mathrm{D}^{(2)})(f))(0)x^2+\frac{1}{6}((\mathrm{D}^{(3)})(f))(0)x^3$$
$$+\frac{1}{24}((\mathrm{D}^{(4)})(f))(0)x^4+\frac{1}{120}((\mathrm{D}^{(5)})(f))(0)x^5+\mathrm{O}(x^6)$$

```
> series(1/(1+x),x);
```

$$1-x+x^2-x^3+x^4-x^5+\mathrm{O}(x^6)$$

```
> series(1/(1+x),x=infinity);
```

$$\frac{1}{x}-\frac{1}{x^2}+\frac{1}{x^3}-\frac{1}{x^4}+\frac{1}{x^5}+\mathrm{O}\left(\frac{1}{x^6}\right)$$

```
> series(1/(1+x),x=-1);
```

$$(x+1)^{(-1)}$$

```
> f:=x*sin(3*x); series(f,x);
```

$$3x^2-\frac{9}{2}x^4+\mathrm{O}(x^6)$$

```
> g:=convert(%,polynom);
```

$$g := 3x^2 - \frac{9}{2}x^4$$

> plot([f,g],x=-1..1,y=-0.5..0.8);

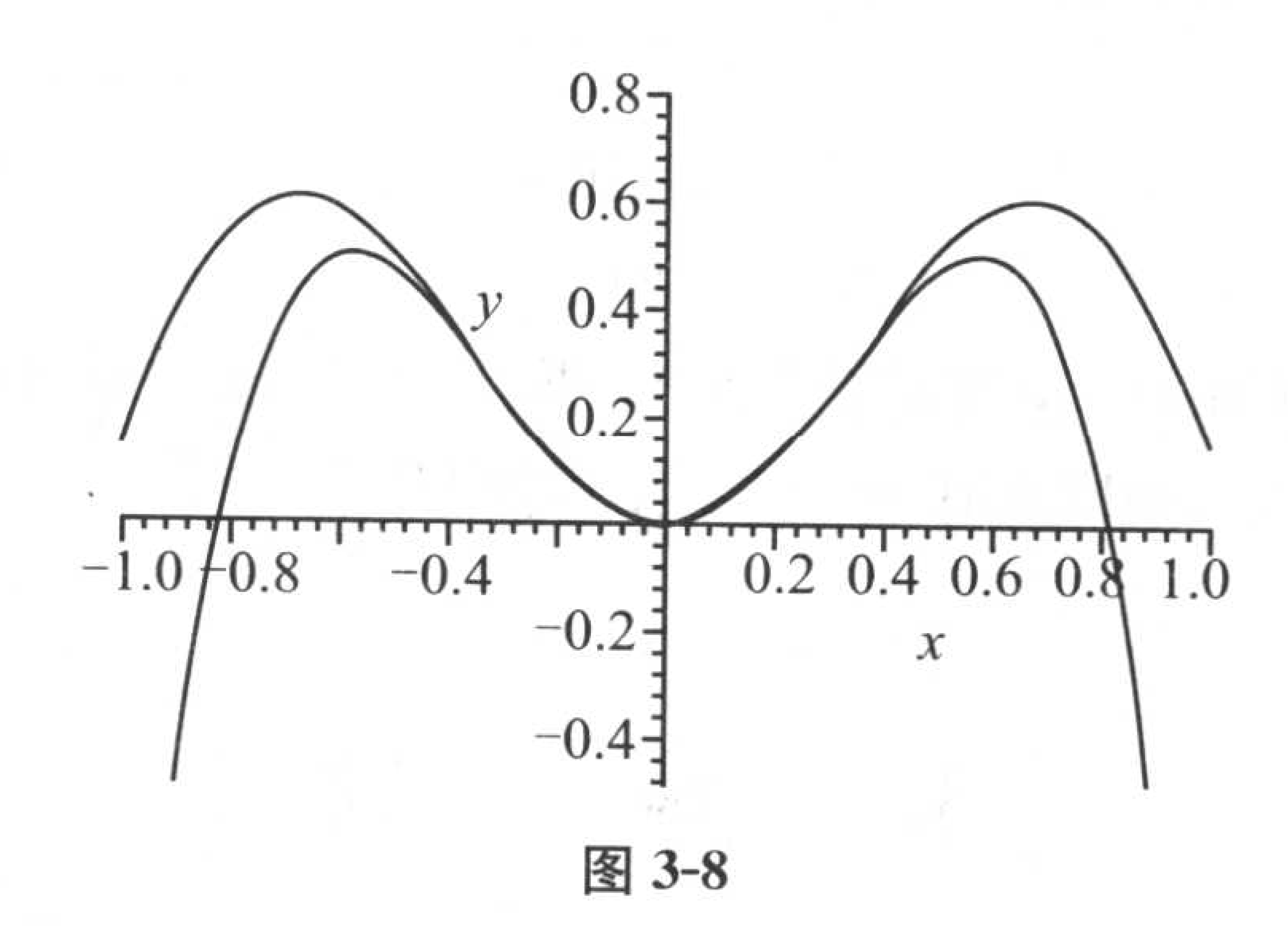


图 3-8

3.5.2 泰勒展开(taylor)

taylor(f(x),x=a,n)

对 $f(x)$在 $x=a$ 处做 n 次泰勒展开。

【例 23】 泰勒级数展开。

> taylor(f(x),x);

$$f(0)+(D(f))(0)x+\frac{1}{2}((D^{(2)})(f))(0)x^2+\frac{1}{6}((D^{(3)})(f))(0)x^3 + \frac{1}{24}((D^{(4)})(f))(0)x^4+\frac{1}{120}((D^{(5)})(f))(0)x^5+O(x^6)$$

> taylor(1/(1+x),x=-1);

Error, does not have a taylor expansion, try series()

3.5.3 多元泰勒展开(mtaylor)

mtaylor(f(x),x=a,n)

对 $f(x)$在 $x=a$ 处做 n 次泰勒展开，其中 x、a 为变量列表或集合，n 为非负整数，缺省值是 6。与 series 函数及 taylor 函数不同的是，mtaylor 的返回值中不含 $O(x^a)$项。

【例 24】 多元泰勒级数展开。

> mtaylor(f(x,y,z),[x,y,z],3);

$$f(0,0,0)+(D_1(f))(0,0,0)x+(D_2(f))(0,0,0)y+(D_3(f))(0,0,0)z$$
$$+\frac{1}{2}(D_{1,1}(f))(0,0,0)x^2+x(D_{1,2}(f))(0,0,0)y+x(D_{1,3}(f))(0,0,0)z$$
$$+\frac{1}{2}(D_{2,2}(f))(0,0,0)y^2+y(D_{2,3}(f))(0,0,0)z+\frac{1}{2}(D_{3,3}(f))(0,0,0)z^2$$

```
> mtaylor(exp(x+y),{x,y},3);
```

$$1+x+y+\frac{1}{2}x^2+yx+\frac{1}{2}y^2$$

```
> mtaylor(sqrt(x^2+y^2),[x,y]);
```

$$\sqrt{x^2+y^2}$$

3.6　积分变换

Maple 的 inttrans 函数包提供了一批有关积分变换的函数，如 Fourier 变换、Laplace变换等。这些变换特别有助于微分方程的求解。

3.6.1　Fourier 变换

```
fourier(f(t),t,w)
invfourier(F(w),w,t)
fouriersin(f(t),t,s)
fouriercos(f(t),t,s)
```

分别计算 Fourier 变换 $F(s)=\int_{-\infty}^{\infty}f(t)\mathrm{e}^{-\mathrm{i}wt}\,\mathrm{d}t$ 及其反变换 $f(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}F(w)\mathrm{e}^{\mathrm{i}st}\,\mathrm{d}w$，Fourier Sine 变换 $F_{\sin}(s)=\sqrt{\frac{2}{\pi}}\int_{-\infty}^{\infty}f(t)\sin(st)\,\mathrm{d}t$ 和 Fourier Cosine 变换 $F_{\cos}(s)=\sqrt{\frac{2}{\pi}}\int_{-\infty}^{\infty}f(t)\cos(st)\,\mathrm{d}t$ 。例如：

```
> with(inttrans);
[addtable, fourier, fouriercos, fouriersin, hankel, hilbert, invfourier, invhilbert,
 invlaplace, invmellin, laplace, mellin, savetable]
> fourier(1+t^2,t,w);
```

$$2\pi(\mathrm{Dirac}(w)-\mathrm{Dirac}(2,w))$$

```
> invfourier(%,w,t);
```

$$1+t^2$$

```
> fourier(1/(1+t^2),t,w);
```

$$\pi(e^{w}\text{Heaviside}(-w)+e^{(-w)}\text{Heaviside}(w))$$

```
> invfourier(%,w,t);
```

$$\frac{1}{1+t^2}$$

3.6.2 Laplace 变换

```
laplace(f(t),t,s)
invlaplace(F(s),s,t)
```

分别计算 Laplace 变换 $L(s)=\int_0^{\infty}f(t)e^{-st}\,dt$ 及其反变换。

```
> with(inttrans): f(t):=t^3*cos(t):
> F(s):=laplace(f(t),t,s);
```

$$F(s):=\frac{6(s^4+1-6s^2)}{(s^2+1)^4}$$

```
> invlaplace(F(s),s,t);
```

$$t^3\cos(t)$$

inttrans 函数包中类似的变换还有 Hilbert 变换 $F(s)=\frac{1}{\pi}\int_{-\infty}^{\infty}\frac{f(t)}{t-s}dt$，Hankel 变换 $H_n(s)=\int_0^{\infty}f(t)\text{BesselJ}(n,t,s)\sqrt{st}\,dt$，Mellin 变换 $M(s)=\int_0^{\infty}f(t)t^{s-1}dt$，以及它们的反变换。

3.6.3 离散变换

离散 Fourier 变换 $y=\left(\frac{1}{\sqrt{n}}e^{\frac{2\pi\sqrt{-1}}{n}ij}\right)_{n\times n}x$ 是离散 Fourier 分析中常用的一种变换。DiscreteTransforms 函数包提供了计算点列的离散 Fourier 变换的函数，

```
FourierTransform(数据,点数,选项)
InverseFourierTransform(数据,点数,选项)
```

其中数据的类型必须为 Vector、Matrix 或 Array。例如：

```
> with(DiscreteTransforms);
```

[FourierTransform, InverseFourierTransform]

```
> x:=Array([1,3,5,7]): y:=FourierTransform(x);
```

$$y:=[8.+0.I,-2.+2.I,-2.+0.I,-2.-2.I]$$

```
> InverseFourierTransform(y);
```

$$[1.+0.I,3.+0.I,5.+0.I,7.+0.I]$$

FourierTransform 函数和 InverseFourierTransform 函数仅适用于数值计算，无法进行符号计算。用户可以在选项中设置变换的算法 algorithm=mintime（快速算法，缺省值），algorithm=minstorage（内存优化算法）或 algorithm=DFT（古典算法）。

Z变换$Z(x)=\sum_{n=0}^{\infty}f(n)x^{-n}$也是离散系统分析中常用的一种变换，是 Laplace 变换的离散形式，主要应用数字信号处理、控制论和解微分方程等离散系统分析中。$Z(x^{-1})$也经常被称为数列$(f(n))_0^{\infty}$的母函数。

```
ztrans(f,n,x)
invztrans(f,x,n)
```

```
> ztrans(1/2^n,n,x);
```

$$\frac{2x}{2x-1}$$

```
> invztrans(%,x,n);
```

$$\left(\frac{1}{2}\right)^n$$

ztrans 和 invtrans 为 Maple 的标准库函数，可直接使用。

习 题

1. 计算极限。

(1) $\lim\limits_{x\to 0}\frac{e^x-e^{-x}}{\sin x}$

(2) $\lim\limits_{x\to\infty}\frac{\sqrt{x}\arctan x-1}{\sqrt{x^2-x}}$

(3) $\lim\limits_{x\to 0}\frac{\sin 3x}{x}$

(4) $\lim\limits_{x\to 0}(1-x)^x$

2. 求函数的微商。

(1) $y=x^2\sin x^2$

(2) $y=\sqrt{x+\sqrt{x+\sqrt{x}}}$

(3) $y=\log_a x+ba^x, a>0$ (4) $y=\frac{\arcsin 2x}{\sqrt{1-x^2}}$

(5) e^x+x^e (6) $y=x^{x^x}, x>0$

(7) $y=(\cos x)^{\sin x}$ (8) $y=\ln\left(\cos\left(\arctan\left(\frac{1}{2}(e^x-e^{-x})\right)\right)\right)$

3. 计算高阶导数。

(1) $y^{(10)}, y=\cos x\cdot\cos 2x\cdot\cos 3x$ (2) $y^{(20)}$, $y=x^2\cos x$

(3) $y^{(60)}, y=\frac{x+2}{1-x^2}$ (4) $y^{(60)}, y=\frac{1+x}{\sqrt{1-x}}$

4. 计算不定积分。

(1) $\int(2x-7)^{2006}\mathrm{d}x$ (2) $\int\left(1-\frac{1}{x}\right)^2\mathrm{d}x$ (3) $\int\frac{1}{\sqrt{a^2+x^2}}\mathrm{d}x, a>0$

(4) $\int\frac{x^2-4x}{x^2-2x-3}\mathrm{d}x$ (5) $\int x^2\arctan x\,\mathrm{d}x$ (6) $\int\frac{e^{2x}+1}{e^x+1}\mathrm{d}x$

(7) $\iint x\cos(xy)\,\mathrm{d}x\mathrm{d}y$ (8) $\iiint xyz(1-x-y)\,\mathrm{d}x\mathrm{d}y\mathrm{d}z$

5. 计算定积分。

(1) $\int_0^1(3x-4x^2)^2\mathrm{d}x$ (2) $\int_0^{\ln 2}\sqrt{e^x-1}\,\mathrm{d}x$ (3) $\int_0^1\frac{\sqrt{e^x}}{\sqrt{e^x+e^{-x}}}\mathrm{d}x$

(4) $\int_0^a\frac{x^2}{\sqrt{x^2+a^2}}\mathrm{d}x$ (5) $\int_0^1\int_0^1 xy^3e^{x^2+y^2}\,\mathrm{d}y\mathrm{d}x$ (6) $\int_0^1\mathrm{d}y\int_y^{\sqrt{y}}\frac{\sin x}{x}\mathrm{d}x$

(7) $\int_0^{2\pi}\mathrm{d}\phi\int_0^a r^2\sin^2\phi\,\mathrm{d}r$ (8) $\int_0^1\mathrm{d}x\int_0^x\mathrm{d}y\int_0^{x+y}xyz\,\mathrm{d}z$

6. 求幂级数在原点处的 5 阶展开式。

(1) e^{x^2} (2) $\frac{x^2}{1-x}$ (3) $\ln\sqrt{\frac{1+x}{1-x}}$

(4) $(1+x)e^{-x}$ (5) $\cos x\cos y$

第 4 章　线 性 代 数

4.1　关于线性代数函数包

Maple 11 中提供了 5 个矩阵和向量运算的函数包 linalg、LinearAlgebra、VectorCalculus、Student[LinearAlgebra]、Student[VectorCalculus]，其中 linalg 基本被 LinearAlgebra 和 VectorCalculus 所取代。在将来的升级 Maple 版本中，linalg 将会被删除，推荐使用 LinearAlgebra 和 VectorCalculus 函数包。

在本章中，我们主要介绍 LinearAlgebra 和 VectorCalculus 函数包中与线性代数相关的函数的用法，Student[LinearAlgebra]和 Student[VectorCalculus]函数包含有教学辅导功能，函数包中的函数虽然形式可能有所不同，但用法类似，故而忽略。

1. 关于 linalg 函数包

```
> with(linalg);
[BlockDiagonal, GramSchmidt, JordanBlock, LUdecomp, QRdecomp, Wronskian, addcol, addrow, adj, adjoint, angle, augment, backsub, band, basis, bezout, blockmatrix, charmat, charpoly, cholesky, col, coldim, colspace, colspan, companion, concat, cond, copyinto, crossprod, curl, definite, delcols, delrows, det, diag, diverge, dotprod, eigenvals, eigenvalues, eigenvectors, eigenvects, entermatrix, equal, exponential, extend, ffgausselim, fibonacci, forwardsub, frobenius, gausselim, gaussjord, geneqns, genmatrix, grad, hadamard, hermite, hessian, hilbert, htranspose, ihermite, indexfunc, innerprod, intbasis, inverse, ismith, issimilar, iszero, jacobian, jordan, kernel, laplacian, leastsqrs, linsolve, matadd, matrix, minor, minpoly, mulcol, mulrow, multiply, norm, normalize, nullspace, orthog, permanent, pivot, potential, randmatrix, randvector, rank, ratform, row, rowdim, rowspace, rowspan, rref, scalarmul, singularvals, smith, stackmatrix, submatrix, subvector, sumbasis, swapcol, swaprow, sylvester,
```

toeplitz, trace, transpose, vandermonde, vecpotent, vectdim, vector, wronskian]

linalg 函数包共有 114 个函数,可做数值计算和符号计算,侧重于抽象的线性代数运算。除了 BlockDiagonal、GramSchmidt、JordanBlock、LUdecomp、QRdecomp、Wronskian 和惰性函数之外,其他函数的名称都为小写字母。

linalg 函数包的常用数据类型为 matrix 和 vector,它们都是由数据类型 array 以及基本数据类型 list 所派生出来的,不同数据类型 list、array、matrix、vector 可混合使用。

在计算中,具有类型 array、matrix、vector 的变量遵循后名计算(Last Name Evaluation)原则,即表达式的值并不被立即计算出来,必须通过 eval、print 或 evalm 才能显示出来。矩阵加减法和数乘仍用 $A\pm B$,λA,乘法要用 $A\&*B$,矩阵方幂或逆矩阵方幂可用 A^n,显示矩阵则用 evalm。例如:

```
> a : =array([[1,2],[3,4]]): a;
```

$$a$$

```
> b : =vector([5,6]): b;
```

$$b$$

```
> c : =a&*b: c;
```

$$a\&*b$$

```
> eval(a),eval(b),evalm(c);
```

$$\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix},[5 \quad 6],[17 \quad 39]$$

2. 关于 LinearAlgebra 和 Student[LinearAlgebra]函数包

```
> with(LinearAlgebra);
```

[&x, Add, Adjoint, BackwardSubstitute, BandMatrix, Basis, BezoutMatrix, BidiagonalForm, BilinearForm, CharacteristicMatrix, CharacteristicPolynomial, Column, ColumnDimension, ColumnOperation, ColumnSpace, CompanionMatrix, ConditionNumber, ConstantMatrix, ConstantVector, Copy, CreatePermutation, CrossProduct, DeleteColumn, DeleteRow, Determinant, Diagonal, DiagonalMatrix, Dimension, Dimensions, DotProduct, EigenConditionNumbers, Eigenvalues, Eigenvectors, Equal, ForwardSubstitute, FrobeniusForm, GaussianElimination, GenerateEquations, GenerateMatrix, Generic, GetResultDataType, GetResultShape, GivensRotationMatrix, GramSchmidt, HankelMatrix, HermiteForm, HermitianTranspose, HessenbergForm, HilbertMatrix, HouseholderMatrix, IdentityMatrix, IntersectionBasis, IsDefinite, IsOrthogonal, IsSimilar, IsUnitary, JordanBlockMatrix, JordanForm, LA_Main, LUDecom-

position, LeastSquares, LinearSolve, Map, Map2, MatrixAdd, MatrixExponential, MatrixFunction, MatrixInverse, MatrixMatrixMultiply, MatrixNorm, MatrixPower, MatrixScalarMultiply, MatrixVectorMultiply, MinimalPolynomial, Minor, Modular, Multiply, NoUserValue, Norm, Normalize, NullSpace, OuterProductMatrix, Permanent, Pivot, PopovForm, QRDecomposition, RandomMatrix, RandomVector, Rank, RationalCanonicalForm, ReducedRowEchelonForm, Row, RowDimension, RowOperation, RowSpace, ScalarMatrix, ScalarMultiply, ScalarVector, SchurForm, SingularValues, SmithForm, StronglyConnectedBlocks, SubMatrix, SubVector, SumBasis, SylvesterMatrix, ToeplitzMatrix, Trace, Transpose, TridiagonalForm, UnitVector, VandermondeMatrix, VectorAdd, VectorAngle, VectorMatrixMultiply, VectorNorm, VectorScalarMultiply, ZeroMatrix, ZeroVector, Zip]

LinearAlgebra 函数包共有 120 个函数，以线性代数的运算为主，它几乎覆盖了 linalg 函数包的所有线性代数功能。LinearAlgebra 的函数命名比 linalg 更加自然，使用更方便，所有函数名都以大写字母开头。

LinearAlgebra 函数包的常用数据类型为 Array、Matrix 和 Vector，它们都基于数据类型 rtable。除了一般形式的矩阵，通过 rtable 还可以表示一些具有特殊形式的矩阵，如对角矩阵、三角矩阵、带状矩阵、稀疏矩阵、结构矩阵等。这样在数值计算、符号计算以及混合计算的时候，尤其是在大规模矩阵计算的时候，LinearAlgebra 的功能要比 linalg 强大和有效的多。

在计算中，LinearAlgebra 严格区分不同数据类型 Array、Matrix、Vector[Row]、Vector[Column]，它们之间不可乱用。矩阵运算的形式则较为简单。矩阵加减法和数乘仍用 $A\pm B$, λA，乘法要用 $A.B$，矩阵方幂或逆矩阵方幂可用 $A\hat{}n$。运算的结果总是立即显示出来。例如：

```
> a := Array([[1,2],[3,4]]);
```

$$a := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

```
> a^2;
```

$$\begin{bmatrix} 1 & 4 \\ 9 & 16 \end{bmatrix}$$

```
> b := Matrix([[1,2],[3,4]]);
```

$$b := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

```
> b^2;
```

$$\begin{bmatrix} 7 & 10 \\ 15 & 22 \end{bmatrix}$$

```
> c：=Vector([5,6])；
```

$$c := \begin{bmatrix} 5 \\ 6 \end{bmatrix}$$

```
> a.c;
```

Error, (in LinearAlgebra：-Multiply) expects its 1st argument, MV1, to be of type {Matrix,Vector,scalar}, but received Array(1..2,1..2, [··· unable to display content ···])

```
> b.c;
```

$$\begin{bmatrix} 17 \\ 39 \end{bmatrix}$$

Student 函数包中的 LinearAlgebra 子函数包专门为线性代数提供了一些交互式的辅助教学程序,如 EigenPlotTutor、EigenvaluesTutor、EigenvectorsTutor、InverseTutor、LinearSolveTutor、LinearSystemPlotTutor、LinearTransformPlotTutor 等。

3. 关于 VectorCalculus 和 Student[VectorCalculus]函数包

```
> with(VectorCalculus);
```

[&x, *, +, −, ., <,>, <|>, About, AddCoordinates, ArcLength, BasisFormat, Binormal, Compatibility, ConvertVector, CrossProd, CrossProduct, Curl, Curvature, D, Del, DirectionalDiff, Divergence, DotProd, DotProduct, Flux, GetCoordinateParameters, GetCoordinates, GetPVDescription, GetRootPoint, GetSpace, Gradient, Hessian, Jacobian, Laplacian, LineInt, MapToBasis, Nabla, Norm, Normalize, PathInt, PlotPositionVector, PlotVector, PositionVector, PrincipalNormal, RadiusOfCurvature, RootedVector, ScalarPotential, SetCoordinateParameters, SetCoordinates, SpaceCurve, SurfaceInt, TNBFrame, Tangent, TangentLine, TangentPlane, TangentVector, Torsion, Vector, VectorField, VectorPotential, VectorSpace, Wronskian, diff, eval, evalVF, int, limit, series]

VectorCalculus 函数包以向量和向量场的计算为主,特别是向量和向量场在不同坐标系下的转化。Student 函数包中的 VectorCalculus 子函数包专门为此提供了一些交互式的辅助教学程序,如 SpaceCurveTutor、VectorFieldTutor 等。

4.2　向量的定义和运算

4.2.1　定 义 向 量

1. 向量 Vector

Vector[类型](维数,初值,选项)

其中类型可为行向量(row)或列向量(column),缺省值 column。关于选项的具体用法,感兴趣的读者请参阅 Maple 的联机帮助、用户手册以及相关的书籍资料。Vector 为标准库函数。

【例 1】 用 Vector 命令定义向量。

```
> u：=Vector([a,b,c])； # 定义列向量
```

$$u := \begin{bmatrix} a \\ b \\ c \end{bmatrix}$$

```
> v：=Vector[row]([a,b,c])； # 定义行向量
```

$$v := [a \quad b \quad c]$$

```
> whattype(u),whattype(v)；
```

$$Vector_{column}, Vector_{row}$$

```
> A：=Vector[row]([1,2,3],readonly=true)； # 定义为只读向量
```

$$A := [1 \quad 2 \quad 3]$$

```
> A[1]：=5； # 不允许修改具有只读属性向量的元素
```

Error, cannot assign to a read-only Vector

```
> Vector[row](5,x-> x^2)； # 用函数定义向量元素
```

$$[1 \quad 4 \quad 9 \quad 16 \quad 25]$$

【例 2】 将列表转化为向量。

```
> u：=[1,2,3]； v：=convert(u,Vector[row])； w：=convert(u,Vector)；
```

$$u := [1, 2, 3]$$

$$v := [1 \quad 2 \quad 3]$$

$$w := \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

> whattype(u),whattype(v),whattype(w);

$$list,\ Vector_{row},\ Vector_{column}$$

2. 随机向量 RandomVector

RandomVector[类型](维数,选项)

其中类型和选项的用法同 Vector。RandomVector 位于 LinearAlgebra 函数包中。

【例 3】 生成随机向量。

> with(LinearAlgebra):

> RandomVector(3); # 生成在－99..99 范围内的 3 维随机列向量

$$\begin{bmatrix} -38 \\ -18 \\ 87 \end{bmatrix}$$

> RandomVector[row](5,generator=1..9); # 生成 1..9 范围内的 3 维随机行向量

[3,5,9,9,4]

本书 1.4.4 节介绍了用 <|> 和 <,> 定义向量的方式;本节介绍了用 Vector 和 RandomVector 函数定义向量的方式;8.5 节还将介绍利用输入模板定义向量的方式;用 convert 函数也可以将其他数据类型转换为向量。

4.2.2 向量运算

1. 加法和数乘

向量的线性运算加减法和数乘的最简单形式为:

U±V, c*U

其中 U,V 为向量,c 为常数。

更一般的形式为:

VectorAdd(U,V,c1,c2,选项)

计算向量 U 和 V 的线性组合 $c1*U+c2*V$。VectorAdd 为 LinearAlgebra 函数包中的函数。

【例4】 向量的线性运算。

```
> a := <1,2,3> ; b := Vector([4,5,6]);
> a+b, LinearAlgebra:-VectorAdd(a,b,1,-1);
```

$$\begin{bmatrix}5\\7\\9\end{bmatrix},\begin{bmatrix}-3\\-3\\-3\end{bmatrix}$$

```
> map(int,a,x) ; # 请观察函数作用到向量的效果
```

$$\begin{bmatrix}x\\2x\\3x\end{bmatrix}$$

2. 内积和外积

U. V 或 DotProduct(U,V)
U &x V 或 CrossProduct(U,V)
BilinearForm(U,V,S)

上述式中运算符“.”为标准库函数，&x、DotProduct、CrossProduct 存在于函数包 LinearAlgebra和 VectorCalculus 中，用法和结果均稍有差别，BilinearForm 存在于函数包 LinearAlgebra 中，S 的缺省值为单位阵。

【例5】 观察在不同函数包中向量的内积和外积。

```
> u := Vector([a,b,c]); v := Vector([x,y,z]); # 定义列向量 u,v
> u.v, LinearAlgebra:-DotProduct(u,v), VectorCalculus:-DotProd(u,v);
```

$$\bar{a}x+\bar{b}y+\bar{c}z,\bar{a}x+\bar{b}y+\bar{c}z,ax+by+cz$$

```
> LinearAlgebra:-CrossProduct(u,v), VectorCalculus:-CrossProd(u,v);
```

$$\begin{bmatrix}bz-cy\\cx-az\\ay-bx\end{bmatrix},(bz-cy)\mathrm{e}_x+(cx-az)\mathrm{e}_y+(ay-bx)\mathrm{e}_z$$

```
> u := Vector[row]([a,b,c]); v := Vector([x,y,z]);
# 定义行向量 u,列向量 v
> u.v, LinearAlgebra:-DotProduct(u,v), VectorCalculus:-DotProd(u,v);
```

$$ax+by+cz,\bar{x}a+\bar{y}b+\bar{z}c,ax+by+cz$$

```
> LinearAlgebra:-CrossProduct(u,v), VectorCalculus:-CrossProd(u,v);
```

同上

```
> u := Vector([a,b,c]); v := Vector[row]([x,y,z]);
# 定义列向量 u,行向量 v
```

> u. v,LinearAlgebra：-DotProduct(u,v),VectorCalculus：-DotProd(u,v);

$$\begin{bmatrix} ax & ay & az \\ bx & by & bz \\ cx & cy & cz \end{bmatrix}, \bar{a}x+\bar{b}y+\bar{c}z, ax+by+cz$$

> LinearAlgebra：-CrossProduct(u,v),VectorCalculus：-CrossProd(u,v);

同上

> u：=Vector[row]([a,b,c])：v：=Vector[row]([x,y,z])：

定义行向量 u,v

> u. v,LinearAlgebra：-DotProduct(u,v),VectorCalculus：-DotProd(u,v);

$$\bar{x}a+\bar{y}b+\bar{z}c, \bar{x}a+\bar{y}b+\bar{z}c, ax+by+cz$$

> LinearAlgebra：-CrossProduct(u,v),VectorCalculus：-CrossProd(u,v);

$$[bz-cy \quad cx-az \quad ay-bx], (bz-cy)e_x+(cx-az)e_y+(ay-bx)e_z$$

3. 长度和角度

Norm(V,p)或 VectorNorm(V,p)
Normalize(V,p)
VectorAngle(U,V)

分别计算向量 V 的 p 范数、单位化和向量 U 和 V 的夹角，其中 Norm、Normalize 存在于函数包LinearAlgebra和 VectorCalculus 中。在函数包 LinearAlgebra 中，p 的缺省值为∞；而在函数包 VectorCalculus 中，p 的缺省值为 2。VectorAngle、VectorNorm 存在于函数包 LinearAlgebra 中。

【例 6】 观察在不同函数包中向量的长度和夹角。

> u：=Vector([a,b,c])：v：=Vector([x,y,z])：

> LinearAlgebra：-Norm(u),VectorCalculus：-Norm(u),
LinearAlgebra：-VectorAngle(u,v);

$$\max(|b|,|a|,|c|), \sqrt{a^2+b^2+c^2}, \arccos\left(\frac{\bar{a}x+\bar{b}y+\bar{c}z}{\sqrt{|a|^2+|b|^2+|c|^2}\sqrt{|x|^2+|y|^2+|z|^2}}\right)$$

4. 向量空间

Basis(V)
SumBasis([V1,…,Vn])
IntersectionBasis([V1,…,Vn])
GramSchmidt(V)

分别计算向量组的极大线性无关组、n 个向量组的极大线性无关组、n 个向量组生成的子空间的交空间的一组基、向量组的 Gram-Schmidt 正交化，其中向量组 $V,V1,\cdots,Vn$ 可为列表或集合。Basis、SumBasis、IntersectionBasis、GramSchmidt 都是函数包 LinearAlgebra 中的函数。

【例 7】 向量组正交化。

```
> with(LinearAlgebra): v1 := <2,1,-1>: v2 := <1,0,2>: v3 := <1,1,1>:
> a := GramSchmidt([v1,v2,v3]);
```

$$a := \left[\begin{bmatrix}2\\1\\-1\end{bmatrix}, \begin{bmatrix}1\\0\\2\end{bmatrix}, \begin{bmatrix}\frac{-4}{15}\\\frac{2}{3}\\\frac{2}{15}\end{bmatrix}\right]$$

```
> b := GramSchmidt({v1,v2,v3},normalized);
```

$$b := \left\{\begin{bmatrix}\frac{1}{5}\sqrt{5}\\0\\\frac{2}{5}\sqrt{5}\end{bmatrix}, \begin{bmatrix}\frac{1}{3}\sqrt{6}\\\frac{1}{6}\sqrt{6}\\-\frac{1}{6}\sqrt{6}\end{bmatrix}, \begin{bmatrix}-\frac{1}{15}\sqrt{30}\\\frac{1}{6}\sqrt{30}\\\frac{1}{30}\sqrt{30}\end{bmatrix}\right\}$$

4.3　矩阵的定义

4.3.1　定 义 矩 阵

1. 矩阵 Matrix

Matrix(行数,列数,初值,选项)

Matrix 为标准库函数，它是定义矩阵的主要函数。通过设置选项，用户还可以定义特殊形式的矩阵。

【例 8】 定义矩阵。

```
> Matrix(2);
```

$$\begin{bmatrix}0 & 0\\0 & 0\end{bmatrix}$$

> Matrix([[1,2,3],[4,5,6]]);

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

> Matrix(2,3,symbol=u);

$$\begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} \\ u_{2,1} & u_{2,2} & u_{2,3} \end{bmatrix}$$

> M:=Matrix(3,(i,j)->10*i+j); # 用函数定义矩阵元素

$$M:=\begin{bmatrix} 11 & 12 & 13 \\ 21 & 22 & 23 \\ 31 & 32 & 33 \end{bmatrix}$$

> M:=Matrix(3,4,[[a,b],[c,d]],fill=-1);

$$M:=\begin{bmatrix} a & b & -1 & -1 \\ c & d & -1 & -1 \\ -1 & -1 & -1 & -1 \end{bmatrix}$$

> M:=Matrix(%,shape=triangular[lower]);

$$M:=\begin{bmatrix} a & 0 & 0 & 0 \\ c & d & 0 & 0 \\ -1 & -1 & -1 & 0 \end{bmatrix}$$

> M:=Matrix(3,[[1],[a,2],[b,c,3]],shape=hermitian); # 定义厄阵

$$M:=\begin{bmatrix} 1 & \bar{a} & \bar{b} \\ a & 2 & \bar{c} \\ b & c & 3 \end{bmatrix}$$

2. 面板输入

点击输入面板中的"矩阵"面板(图 4-1),选定矩阵的行数、列数、类型、形状和数据类型,再点击"插入矩阵"按钮,在工作窗口的光标处插入矩阵。例如:

$$A=\begin{bmatrix} m_{1,1} & m_{1,2} \\ m_{2,1} & m_{2,2} \end{bmatrix}$$

此时,用户可以修改矩阵的元素值。当行数或列数等于 1 时,用户可以选择插入矩阵或插入向量。虽然表达式一模一样,实际上它们具有不同的变量类型。

3. 上下文菜单

用鼠标右键单击矩阵,弹出如图 4-2 所示的上下文菜单。在此菜单中(例如菜单项"Eigenvalues,etc","Queries","Solvers and Forms","Standard Operartions"),用户可以发现一些有用的矩阵命令。

图 4-1

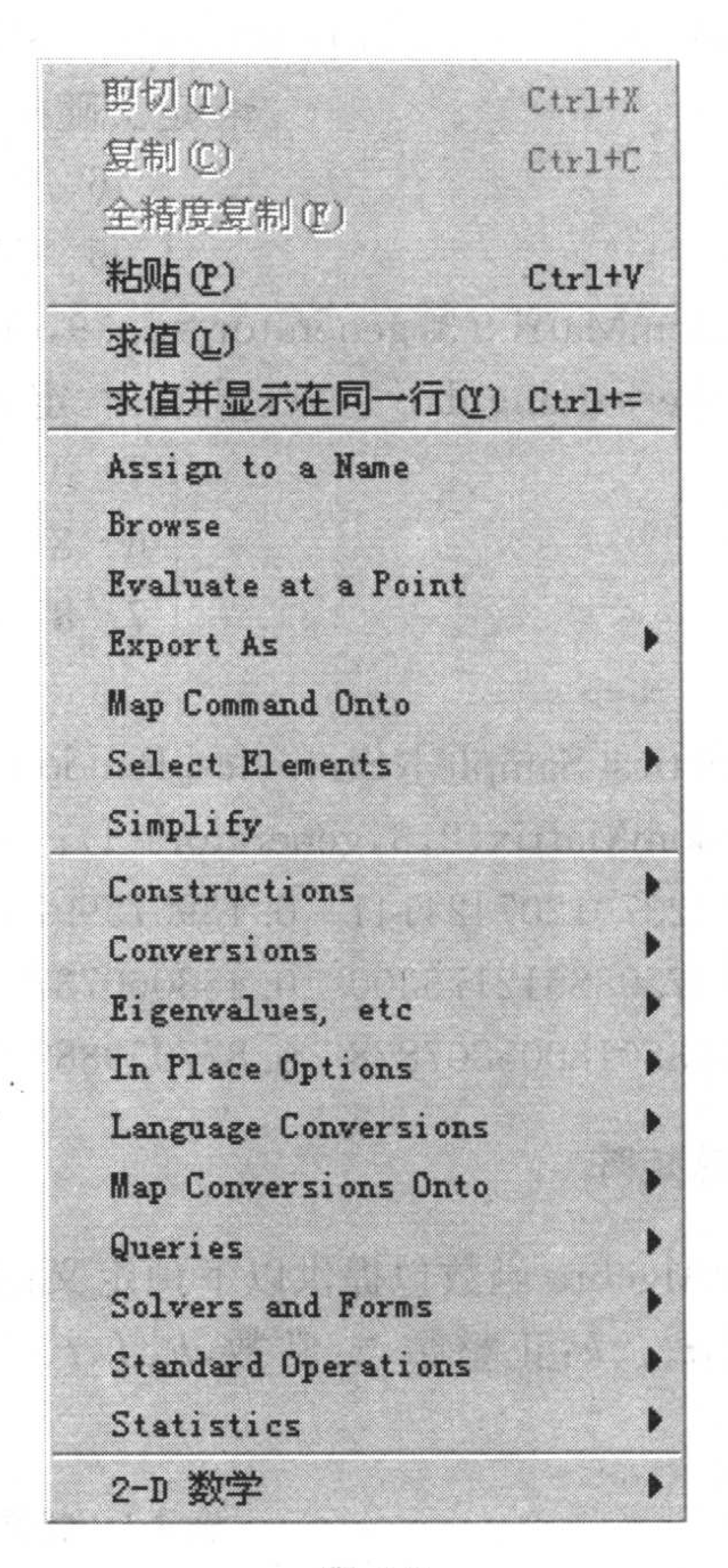


图 4-2

4. 随机矩阵 RandomMatrix

RandomMatrix(行数,列数,选项)

【例 9】 生成随机矩阵。

> with(LinearAlgebra): radomize():

　# 在使用 RandomMatrix 之前,先调入函数包,并初始化随机数发生器

> RandomMatrix(2);

$$\begin{bmatrix} 44 & -31 \\ 92 & 67 \end{bmatrix}$$

> RandomMatrix(2,generator=0..1.0); # 矩阵元素在[0,1]上随机均匀分布

$$\begin{bmatrix} 0.878441803808296728 & 0.142743834910262324 \\ 0.889664198876174206 & 0.986313501353820566 \end{bmatrix}$$

```
> RandomMatrix(2,outputoptions=[shape=symmetric]);
                                    # 定义随机对称阵
```

$$\begin{bmatrix} 46 & 95 \\ 95 & 54 \end{bmatrix}$$

```
> RandomMatrix(3,generator=1..9,outputoptions=
  [shape=triangular[lower]]);  # 定义随机下三角阵
```

$$\begin{bmatrix} 7 & 0 & 0 \\ 6 & 8 & 0 \\ 7 & 8 & 4 \end{bmatrix}$$

```
> f := x ->
  Statistics[Sample](Statistics[RandomVariable](Normal(0,1)),1)[1]:
  RandomMatrix(2,3,generator=f);  # 矩阵元素服从标准正态分布
```

$$\begin{bmatrix} 0.0504123701207424141 & 0.139575997670809248 & 0.198913705440137450 \\ 1.71475988612176300 & 0.458060737299529452 & 0.475421270150344132 \\ -1.29835018005807878 & 0.877059889022590022 & 0.182677608167200523 \end{bmatrix}$$

5. 特殊矩阵

LinearAlgebra 函数包提供以下预定义的特殊矩阵(表 4-1),其中 r:行数,c:列数,d:维数,i、j、k:正整数,s:常数,h、J、t、v:向量或列表,p、q:多项式,x:多项式变元。

表 4-1 特殊矩阵

函　数	说　明
BandMatrix(v,k,r,c)	带状阵
BezoutMatrix(p,q,x)	两个多项式的 Bezout 矩阵多项式按降次排列
CharacteristicMatrix(A,λ)	$\lambda I - A$
CompanionMatrix(p,x)	多项式的友阵
ConstantMatrix(s,r,c)	常值矩阵
DiagonalMatrix(v,r,c)	对角阵
GivensRotationMatrix(v,i,j)	Givens 旋转矩阵
HankelMatrix(h,r)	Hankel 矩阵 $m_{i,j}=h_{i+j}$
HilbertMatrix(r,c,s)	Hilbert 矩阵 $m_{i,j}=\frac{1}{i+j-s}$
HouseholderMatrix(v,d)	Householder 反射矩阵

续表

函　　数	说　　明
IdentityMatrix(r,c)	单位阵 I
JordanBlockMatrix(J,d)	Jordan 标准形
OuterProductMatrix(u,v)	向量的张量积 $m_{i,j}=u_iv_j$
ScalarMatrix(s,r,c)	数量阵 $s\cdot I$
SylvesterMatrix(p,q,x)	两个多项式的 Sylvester 矩阵，多项式按降次排列
ToeplitzMatrix(t,r)	Toeplitz 矩阵 $m_{i,j}=t_{i-j}$
VandermondeMatrix(v,r,c)	Vandermonde 矩阵 $m_{i,j}=v_i^{j-1}$
ZeroMatrix(r,c)	零矩阵 O

【例 10】 定义特殊矩阵。

```
> with(LinearAlgebra):
> convert([[a,b],[1,2]],Matrix); # 将 list 转化为 Matrix
```

$$\begin{bmatrix} a & b \\ 1 & 2 \end{bmatrix}$$

```
> BandMatrix([[w,w],[x,x,x],[y,y,y],[z,z]],1,3,4);
          # 定义 3 行 4 列，起始位置在 1 的带状矩阵
```

$$\begin{bmatrix} x & y & z & 0 \\ w & x & y & z \\ 0 & w & x & y \end{bmatrix}$$

```
> CompanionMatrix(x^3+sum(a[i]*x^i,i=0..2),x);
```

$$\begin{bmatrix} 0 & 0 & -a_0 \\ 1 & 0 & -a_1 \\ 0 & 1 & -a_2 \end{bmatrix}$$

```
> DiagonalMatrix([1,2,3]); # 定义对角线元素是 1,2,3 的矩阵
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

```
> SylvesterMatrix(sum(a[i]*x^i,i=0..3),sum(b[i]*x^i,i=0..3),x);
```

$$\begin{bmatrix} a_3 & a_2 & a_1 & a_0 & 0 & 0 \\ 0 & a_3 & a_2 & a_1 & a_0 & 0 \\ 0 & 0 & a_3 & a_2 & a_1 & a_0 \\ b_3 & b_2 & b_1 & b_0 & 0 & 0 \\ 0 & b_3 & b_2 & b_1 & b_0 & 0 \\ 0 & 0 & b_3 & b_2 & b_1 & b_0 \end{bmatrix}$$

```
> VandermondeMatrix([u,v,w]); # 形成 Vandermonde 矩阵
```

$$\begin{bmatrix} 1 & u & u^2 \\ 1 & v & v^2 \\ 1 & w & w^2 \end{bmatrix}$$

4.3.2　抽取与合成

矩阵的抽取与合成的函数见表 4-2。

表 4-2　矩阵的抽取与合成

函　　数	说　　明
A[i,j]或 SubMatrix(A,i,j)	矩阵元素 a_{ij} 或子向量或子矩阵
〈A,…,B〉	按行构造 rtable 对象
〈A|…|B〉	按列构造 rtable 对象
Diagonal(A,L)	A 的某些对角线构成的列向量序列
Row(A,L)	A 的某些行向量构成的序列
Column(A,L)	A 的某些列向量构成的序列
DeleteRow(A,L)	删除 A 的某些行
DeleteColumn(A,L)	删除 A 的某些列

运算符[ ]、< | > 、<,> 为标准库函数，Diagonal、Row、Column、DeleteRow、DeleteColumn 为函数包 LinearAlgebra 中的函数。表 4-2 中 A,B 为矩阵,L 为集合。

【例 11】 抽取矩阵的部分元素。

```
> A:=Matrix(3,4,(i,j)->a[i,j]);
```

$$A := \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \end{bmatrix}$$

```
> A[-1,1..4]; # 或 Row(A,-1),A 的最后一行
```

$$[a_{3,1}\ \ a_{3,2}\ \ a_{3,3}\ \ a_{3,4}]$$

```
> whatType(%);
```

$$Vector_{row}$$

```
> with(LinearAlgebra):
> Row(A,[2,1]); # A 的第二行和第一行
```

$$[a_{2,1}\ \ a_{2,2}\ \ a_{2,3}\ \ a_{2,4}],[a_{1,1}\ \ a_{1,2}\ \ a_{1,3}\ \ a_{1,4}]$$

```
> DeleteRow(A,[1,2]); # 删除 A 的前两行
```

$$[a_{3,1}\ \ a_{3,2}\ \ a_{3,3}\ \ a_{3,4}]$$

```
> whatType(%);
```

$$Matrix$$

```
> SubMatrix(A,2..3,3..4);
```

$$\begin{bmatrix} a_{2,3} & a_{2,4} \\ a_{3,3} & a_{3,4} \end{bmatrix}$$

```
> Diagonal(A,-2..3); # A 的 6 条对角线
```

$$[a_{3,1}],\begin{bmatrix} a_{2,1} \\ a_{3,2} \end{bmatrix},\begin{bmatrix} a_{1,1} \\ a_{2,2} \\ a_{3,3} \end{bmatrix},\begin{bmatrix} a_{1,2} \\ a_{2,3} \\ a_{3,4} \end{bmatrix},\begin{bmatrix} a_{1,3} \\ a_{2,4} \end{bmatrix},[a_{1,4}]$$

```
> Diagonal(A,outputoptions=[orientation=row]);
```

$$[a_{1,1}\ \ a_{2,2}\ \ a_{3,3}]$$

```
> DiagonalMatrix(Diagonal(A));
```

$$\begin{bmatrix} a_{1,1} & 0 & 0 \\ 0 & a_{2,2} & 0 \\ 0 & 0 & a_{3,3} \end{bmatrix}$$

```
> unwith(LinearAlgebra);
> A[2..3,3..4] := Matrix(2,2,(i,j)->b[i+1,j+2]);
```

$$A_{2..3,3..4} := \begin{bmatrix} b_{2,3} & b_{2,4} \\ b_{3,3} & b_{3,4} \end{bmatrix}$$

```
> A;
```

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & b_{2,3} & b_{2,4} \\ a_{3,1} & a_{3,2} & b_{3,3} & b_{3,4} \end{bmatrix}$$

【例 12】 矩阵合成。

```
> A,B := Matrix([[1,2],[3,4]]),<<a|b>,<c|d>>;
```

$$A, B := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}, \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

><A|B>；# 或 Matrix([A,B])，按行合并矩阵

$$\begin{bmatrix} 1 & 2 & a & b \\ 3 & 4 & c & d \end{bmatrix}$$

><A,B>；# 或 Matrix([[A],[B]])，按列合并矩阵

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \\ a & b \\ c & d \end{bmatrix}$$

> <<A|B>,<B|A>>；# 或 Matrix([[A,B],[B,A]])

$$\begin{bmatrix} 1 & 2 & a & b \\ 3 & 4 & c & d \\ a & b & 1 & 2 \\ c & d & 3 & 4 \end{bmatrix}$$

4.4 矩阵的基本运算

4.4.1 加法和乘法

矩阵的基本运算符和函数见表 4-3。

表 4-3 矩阵的基本运算符和函数

运算符	函　数	说　明
A+B	Add(A,B)	矩阵加法
A . B	Multiply(A,B)	矩阵乘法
a * B 或 A * b	Multiply(a,B)或 Multiply(A,b)	数乘或向量乘
A^n	MatrixPower(A,n)	矩阵方幂
A^(−1)	MatrixInverse(A)	逆矩阵
A^%T	Transpose(A)	转置矩阵
A^%H	HermitianTranspose(A)	共轭转置矩阵
	Adjoint(A)	伴随矩阵

运算符 + ， . ， * ， ^ ， ^%T ， ^%H 为标准库函数，Add、Multiply、MatrixPower、MatrixInverse、Transpose、HermitianTranspose、Adjoint 为函数包 LinearAlgebra中的函数。

【例 13】 计算逆矩阵、伴随矩阵等。

```
> with(LinearAlgebra)：A：=<<a|b>,<c|d>>；
```

$$A:=\begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

```
> A^(-1)；# 或 MatrixInverse(A) # 计算 A 的逆矩阵
```

$$\begin{bmatrix} \frac{d}{-bc+da} & -\frac{b}{-bc+da} \\ -\frac{c}{-bc+da} & \frac{a}{-bc+da} \end{bmatrix}$$

```
> A^(-2), MatrixPower(A,-2)；# 比较结果的不同
```

输出略

```
> A^%T, A^%H；# 计算 A 的转置、共轭转置矩阵
```

$$\begin{bmatrix} a & c \\ b & d \end{bmatrix}, \begin{bmatrix} \bar{a} & \bar{c} \\ \bar{b} & \bar{d} \end{bmatrix}$$

```
> B：=Adjoint(A)；# 计算伴随矩阵
```

$$B:=\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

```
> A . B;
```

$$\begin{bmatrix} -bc+da & 0 \\ 0 & -bc+da \end{bmatrix}$$

4.4.2 初等变换

```
RowOperation(A,L,s)
ColumnOperation(A,L,s)
Pivot(A,i,j)
```

分别作矩阵 A 的行变换、列变换、行消元。例如：

RowOperation(A,[i,j])	交换 A 的第 i 行与第 j 行
RowOperation(A,i,s)	将 A 的第 i 行倍乘 s
RowOperation(A,[i,j],s)	将 A 的第 j 行倍乘 s 加到第 i 行

RowOperation、ColumnOperation、Pivot 都是函数包 LinearAlgebra 中的函数，

具体用法见下例。

【例 14】 矩阵的初等变换。

```
> with(LinearAlgebra): A:=<<1|2|3>,<4|5|6>,<7|8|9>>;
```

$$A := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
> RowOperation(A,[1,3],s); # 第3行乘s加到第1行
```

$$\begin{bmatrix} 1+7s & 2+8s & 3+9s \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
> RowOperation(A,[1,3]); # 互换第1行与第3行
```

$$\begin{bmatrix} 7 & 8 & 9 \\ 4 & 5 & 6 \\ 1 & 2 & 3 \end{bmatrix}$$

```
> RowOperation(A,1,3); # 第1行乘3
```

$$\begin{bmatrix} 3 & 6 & 9 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

```
> Pivot(A,1,3); # 利用第1行a1,3对其他行的第3列"打洞"
```

$$\begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 0 \\ 4 & 2 & 0 \end{bmatrix}$$

```
> RowOperation(A,[1,3],inplace): A; # 互换第1行与第3行,并赋给A
```

$$\begin{bmatrix} 7 & 8 & 9 \\ 4 & 5 & 6 \\ 1 & 2 & 3 \end{bmatrix}$$

```
> A:=Matrix(5,5,(i,j)->a[i,j]);
```

$$A := \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} & a_{1,5} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} & a_{3,5} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} \\ a_{5,1} & a_{5,2} & a_{5,3} & a_{5,4} & a_{5,5} \end{bmatrix}$$

```
> Pivot(A,2,3,[1,3,5]); # 利用第2行的a2,3对第1,3,5行的第3列"打洞"
```

$$\begin{bmatrix} a_{1,1}-\frac{a_{1,3}a_{2,1}}{a_{2,3}} & a_{1,2}-\frac{a_{1,3}a_{2,2}}{a_{2,3}} & 0 & a_{1,4}-\frac{a_{1,3}a_{2,4}}{a_{2,3}} & a_{1,5}-\frac{a_{1,3}a_{2,5}}{a_{2,3}} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} & a_{2,5} \\ a_{3,1}-\frac{a_{3,3}a_{2,1}}{a_{2,3}} & a_{3,2}-\frac{a_{3,3}a_{2,2}}{a_{2,3}} & 0 & a_{3,4}-\frac{a_{3,3}a_{2,4}}{a_{2,3}} & a_{3,5}-\frac{a_{3,3}a_{2,5}}{a_{2,3}} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} & a_{4,5} \\ a_{5,1}-\frac{a_{5,3}a_{2,1}}{a_{2,3}} & a_{5,2}-\frac{a_{5,3}a_{2,2}}{a_{2,3}} & 0 & a_{5,4}-\frac{a_{5,3}a_{2,4}}{a_{2,3}} & a_{5,5}-\frac{a_{5,3}a_{2,5}}{a_{2,3}} \end{bmatrix}$$

4.4.3 矩阵函数

常用的矩阵函数见表 4-4。

表 4-4 常用矩阵函数

函 数	说 明
CharacteristicPolynomial(A,x)	特征多项式 det(xI－A)
ColumnDimension(A)	列数
ColumnSpace(A)	列向量空间的一组基
ConditionNumber(A,p)	取 p 范数时 A 的条件数
Determinant(A)	行列式
Dimension(A)	行数和列数
EigenConditionNumbers(A)	特征值条件数
Eigenvalues(A)	特征值
Eigenvectors(A)	特征向量
Equal(A,B)	矩阵是否相等
IsDefinite(A)	是否定正矩阵
IsOrthogonal(A)	是否正交矩阵
IsSimilar(A,B)	矩阵是否相似
IsUnitary(A)	是否酉矩阵

续表

函　　数	说　　明
MinimalPolynomial(A,x)	极小多项式
Minor(A,r,c)	余子式
Norm(A,p)或 MatrixNorm(A,p)	p-范数
NullSpace(A)	零空间的一组基
Permanent(A)	积和式
Rank(A)	秩
RowDimension(A)	行数
RowSpace(A)	行向量空间的一组基
SingularValues(A)	奇异值
Trace(A)	迹

1. 行列式

【例 15】 计算行列式。

```
> with(LinearAlgebra): A:=<<a|b|c>,<f|g|h>,<p|q|r>>;
```

$$A := \begin{bmatrix} a & b & c \\ f & g & h \\ p & q & r \end{bmatrix}$$

```
> Determinant(A); # 计算矩阵 A 的行列式
```

$$agr-ahq+fqc-fbr+pbh-pgc$$

```
> Permanent(A); # 计算矩阵 A 的积和式
```

$$agr+ahq+fbr+fqc+pbh+pgc$$

```
> Matrix(3,(i,j)->Minor(A,i,j)); # 与伴随矩阵比较
```

$$\begin{bmatrix} gr-hq & fr-hp & fq-gp \\ br-qc & ar-cp & aq-bp \\ bh-gc & ah-cf & ag-bf \end{bmatrix}$$

```
> Adjoint(A);
```

$$\begin{bmatrix} gr-hq & -br+qc & bh-gc \\ -fr+hp & ar-cp & -ah+cf \\ fq-gp & -aq+bp & ag-bf \end{bmatrix}$$

【例 16】 计算球坐标(图 4-3)的 Jacobian 行列式。

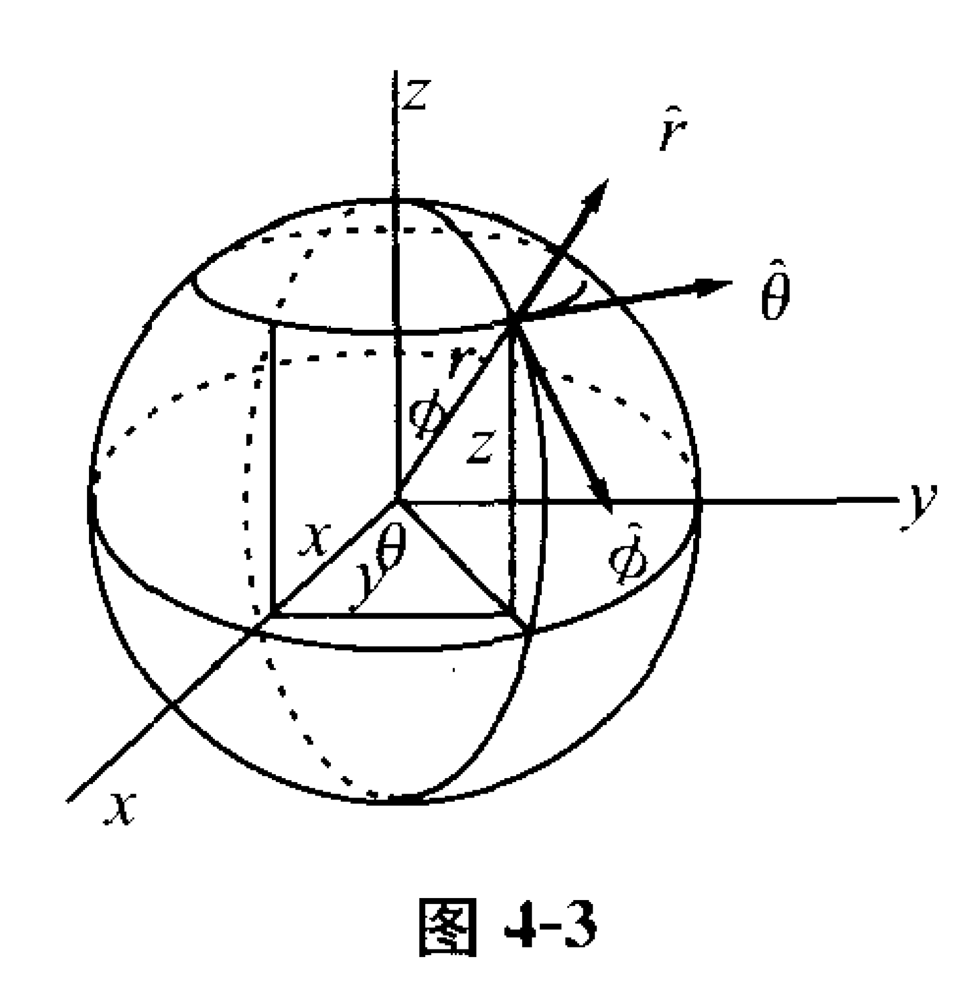


图 4-3

```
> with(LinearAlgebra):
> v := [r * sin(phi) * cos(theta), r * sin(phi) * sin(theta), r * cos(phi)];
```

$$v := [r\sin(\phi)\cos(\theta), r\sin(\phi)\sin(\theta), r\cos(\phi)]$$

```
> J := Matrix([diff(v,r), diff(v,phi), diff(v,theta)]);
```

$$J := \begin{bmatrix} \sin(\phi)\cos(\theta) & \sin(\phi)\sin(\theta) & \cos(\phi) \\ r\cos(\phi)\cos(\theta) & r\cos(\phi)\sin(\theta) & -r\sin(\phi) \\ -r\sin(\phi)\sin(\theta) & r\sin(\phi)\cos(\theta) & 0 \end{bmatrix}$$

```
> Determinant(J);
```

$$\sin(\phi)^3\cos(\theta)^2r^2 + r^2\cos(\phi)^2\cos(\theta)^2\sin(\phi) + r^2\sin(\phi)^3\sin(\theta)^2 + r^2\sin(\phi)\sin(\theta)^2\cos(\phi)^2$$

```
> simplify(%);
```

$$\sin(\phi)r^2$$

2. 行(列)向量空间

【例 17】 向量空间的基、维数。

```
> with(LinearAlgebra): A := <<1,2,0>|<0,2,6>|<0,0,4>|<0,0,0>>;
```

$$A := \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 2 & 0 & 0 \\ 0 & 6 & 4 & 0 \end{bmatrix}$$

```
> [Dimension(A)],[RowDimension(A)],[ColumnDimension(A)];
```

$$[3,4],[3],[4]$$

```
> RowSpace(A); # 行向量空间的一组基
```

$$[[1\ \ 0\ \ 0\ \ 0],[0\ \ 1\ \ 0\ \ 0],[0\ \ 0\ \ 1\ \ 0]]$$

```
> ColumnSpace(A); # 列向量空间的一组基
```

$$\left[\begin{bmatrix}1\\0\\0\end{bmatrix},\begin{bmatrix}0\\1\\0\end{bmatrix},\begin{bmatrix}0\\0\\1\end{bmatrix}\right]$$

```
> NullSpace(A); # Ax=0 的解空间的一组基
```

$$\left\{\begin{bmatrix}0\\0\\0\\1\end{bmatrix}\right\}$$

```
> Rank(A);
```

$$3$$

3. 特征值和特征向量

【例 18】 计算矩阵的特征值、特征向量。

```
> with(LinearAlgebra): A := <<1,2,1>|<-1,2,1>|<0,4,2>>;
```

$$A := \begin{bmatrix}1 & -1 & 0\\2 & 2 & 4\\1 & 1 & 2\end{bmatrix}$$

```
> CharacteristicPolynomial(A,x);
```

$$x^3-5x^2+6x$$

```
> factor(%);
```

$$x(x-2)(x-3)$$

```
> Eigenvalues(A,output=list);
```

$$[0,3,2]$$

```
> Trace(A); # 计算矩阵的迹
```

$$5$$

```
> (r,v) := Eigenvectors(A); # 特征向量为输出矩阵的列向量
```

$$\begin{bmatrix}3\\0\\2\end{bmatrix},\begin{bmatrix}-1 & -1 & -2\\2 & -1 & 2\\1 & 1 & 1\end{bmatrix}$$

【例 19】 判别相似矩阵。

```
> B := <<0,1,1>|<0,2,2>|<0,0,3>>;
```

$$B := \begin{bmatrix}0 & 0 & 0\\1 & 2 & 0\\1 & 2 & 3\end{bmatrix}$$

```
> (q,P) := IsSimilar(A,B,output=['query','C']);
```

$$q,P:=\text{true},\begin{bmatrix}0 & 2 & -4\\ -2 & -1 & 0\\ 1 & -1 & 0\end{bmatrix}$$

```
> Equal(P.A,B.P);# 或 Equal(A,p^(-1).B.P)
```

true

```
> SingularValues(A,output=list);
```

$$[0,\sqrt{30},\sqrt{2}]$$

```
> H:=HilbertMatrix(3);
```

$$H:=\begin{bmatrix}1 & \frac{1}{2} & \frac{1}{3}\\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4}\\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5}\end{bmatrix}$$

```
> IsDefinite(H);
```

true

```
> Eigenvalues(H),SingularValues(H);
```

输出略

```
> Eigenvalues(H*1.0),SingularValues(H*1.0); # 比较结果之不同
```

$$\begin{bmatrix}1.40831892712365359+0.\,I\\ 0.122327065853905653+0.\,I\\ 0.00268734035577353116+0.\,I\end{bmatrix},\begin{bmatrix}1.40831892712365403\\ 0.122327065853905959\\ 0.00268734035577351946\end{bmatrix}$$

4. 范数和条件数

【例20】 计算矩阵的范数和条件数。

```
> with(LinearAlgebra): A:=<<1,1,1>|<1,1,-2>|<1,-1,0>>;
```

$$A:=\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & -1\\ 1 & -2 & 0\end{bmatrix}$$

```
> B:=A^(-1);
```

$$B:=\begin{bmatrix}\frac{1}{3} & \frac{1}{3} & \frac{1}{3}\\ \frac{1}{6} & \frac{1}{6} & \frac{-1}{3}\\ \frac{1}{2} & \frac{-1}{2} & 0\end{bmatrix}$$

> Norm(A),Norm(B),ConditionNumber(A); # 等价于 Norm(* ,infinity)

$$3,1,3$$

> Norm(A,1),Norm(B,1),ConditionNumber(A,1);

$$4,1,4$$

> Norm(A,2),Norm(B,2),ConditionNumber(A,2); # 等价于 Norm(* , Euclidean)

$$\sqrt{6},\frac{1}{2}\sqrt{2},\frac{1}{2}\sqrt{2}\sqrt{6}$$

> Norm(A,Frobenius),Norm(B,Frobenius),ConditionNumber(A,Frobenius);

$$\sqrt{11},1,\sqrt{11}$$

> SingularValues(A);

$$[\sqrt{2},\sqrt{3},\sqrt{6}]$$

4.4.4 矩阵分解和标准形

常见的矩阵分解函数和标准形函数见表 4-5。

表 4-5 常用的矩阵分解函数和标准形函数

函　　数	说　　明	
BidiagonalForm(A)	$A=U.B.Vt$	U、Vt 酉阵，B 二对角
FrobeniusForm(A) RationalCanonicalForm(A)	$A=Q.F.Q^{-1}$	F 友阵
GaussianElimination(A)	返回 LUDecomposition 中的 U	
HermiteForm(A,x)	$A=U^{-1}.H$	U 幺模阵，H 上三角
HessenbergForm(A)	$A=Q.H.Q^{-1}$	Q 酉阵，H:Hessenberg
JordanForm(A)	$A=P.J.P^{-1}$	J:Jordan 标准形
LUDecomposition(A)	$A=P.L.U$ $U=U1.R$	P 置换阵，L 下三角，U、$U1$ 上三角，R 上三角阶梯形
PopovForm(A,x)	$A=P.U^{-1}$	U 幺模阵，deg(P)最小

续表

函 数	说 明	
QRDecomposition(A)	$A=Q.R$	Q 酉阵，R 上三角
ReducedRowEchelonForm(A)	返回 LUDecomposition 中的 R	
SchurForm(A)	$A=Q.T.Q^{-1}$	Q 酉阵，T 准上三角
SingularValues(A)	$A=U.S.Vt$	U、Vt 酉阵，S 对角阵
SmithForm(A)	$A=U^{-1}.S.V^{-1}$	U、V 幺模阵，S 对角阵
TridiagonalForm(A)	$A=Q.T.Q^{-1}$	Q 酉阵，T 三对角

【例 21】 矩阵分解。

```
> with(LinearAlgebra):
> A:=<<0|-1|6>,<1|2|1>,<-2|-1|-5>>;
```

$$A:=\begin{bmatrix}0 & -1 & 6\\ 1 & 2 & 1\\ -2 & -1 & -5\end{bmatrix}$$

```
> (P,L,U):= LUDecomposition(A);
```

$$P,L,U:=\begin{bmatrix}0 & 1 & 0\\ 1 & 0 & 0\\ 0 & 0 & 1\end{bmatrix},\begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ -2 & -3 & 1\end{bmatrix},\begin{bmatrix}1 & 2 & 1\\ 0 & -1 & 6\\ 0 & 0 & 15\end{bmatrix}$$

```
> P. L. U; # 验证 A=P. L. U
```

$$\begin{bmatrix}0 & -1 & 6\\ 1 & 2 & 1\\ -2 & -1 & -5\end{bmatrix}$$

```
> A:=<<1|3>,<2|4>,<2|5>>;
```

$$A:=\begin{bmatrix}1 & 3\\ 2 & 4\\ 2 & 5\end{bmatrix}$$

```
> (Q,R):=QRDecomposition(A,fullspan);
```

$$Q,R:=\begin{bmatrix}\frac{1}{3} & \frac{2}{3} & \frac{2}{3}\\ \frac{2}{3} & \frac{-2}{3} & \frac{1}{3}\\ \frac{2}{3} & \frac{1}{3} & \frac{-2}{3}\end{bmatrix},\begin{bmatrix}3 & 7\\ 0 & 1\\ 0 & 0\end{bmatrix}$$

```
> QRDecomposition(A); # 紧缩形式的 QR 分解
```

$$Q,R:=\begin{bmatrix}\frac{1}{3} & \frac{2}{3}\\ \frac{2}{3} & \frac{-2}{3}\\ \frac{2}{3} & \frac{1}{3}\end{bmatrix},\begin{bmatrix}3 & 7\\ 0 & 1\end{bmatrix}$$

```
> Q.R: # 验证 A=Q.R
```

$$\begin{bmatrix}1 & 3\\ 2 & 4\\ 2 & 5\end{bmatrix}$$

```
> U,S,V:=SingularValues(A,output=['U','S','Vt']);
```

$$U,S,V:=$$

$$\begin{bmatrix}-.4110075475364061 15 & -.6217944607534424 68 & -.6666666666666666 30\\ -.5817503338895685 38 & 0.7419268413450644 81 & -.3333333333333333 15\\ -.7018827144811904 39 & -.2508310400809102 84 & 0.6666666666666667 40\end{bmatrix},$$

$$\begin{bmatrix}7.671183818789534 92\\ 0.3910739295090156 50\\ 0.\end{bmatrix},$$

$$\begin{bmatrix}-.3882417257403013 58 & -.9215575741071160 29\\ 0.9215575741071160 29 & -.3882417257403013 58\end{bmatrix},$$

```
> U.DiagonalMatrix(S[1..2],3,2).V;
```

$$\begin{bmatrix}0.9999999999999999 66 & 3.\\ 2. & 4.0000000000000000 88\\ 2. & 5.0000000000000000 88\end{bmatrix}$$

```
> A:=<<2|-1|1>,<2|2|-1>,<1|2|-1>>;
```

$$A:=\begin{bmatrix}2 & -1 & 1\\ 2 & 2 & -1\\ 1 & 2 & -1\end{bmatrix}$$

```
> sort(CharacteristicPolynomial(A,x));
```

$$x^3-3x^2+3x-1$$

```
> FrobeniusForm(A);
```

$$\begin{bmatrix}0 & 0 & 1\\ 1 & 0 & -3\\ 0 & 1 & 3\end{bmatrix}$$

```
> SmithForm(x-A);
```

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & (-1+x)(1-2x+x^2) \end{bmatrix}$$

```
> Q,J:=JordanForm(A,output=['Q','J']);
```

$$Q,J:=\begin{bmatrix} 0 & 1 & 1 \\ 3 & 2 & 0 \\ 3 & 1 & 0 \end{bmatrix},\begin{bmatrix} 1 & 1 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}$$

```
> Q.J.Q^(-1);
```

$$\begin{bmatrix} 2 & -1 & 1 \\ 2 & 2 & -1 \\ 1 & 2 & -1 \end{bmatrix}$$

4.5　线性方程组

4.5.1　方程与矩阵的转换

对于给定的矩阵 A、B，GenerateEquations 函数可以构造线性方程组 $AX=B$。对于给定的线性方程组，GenerateMatrix 函数可以提取方程组的系数矩阵，将方程组转化为矩阵乘积 $AX=B$ 的形式。GenerateEquations 和 GenerateMatrix 都是 LinearAlgebra函数包中的函数。它们的用法有：

GenerateEquations(A,变量列表,B)
GenerateMatrix(方程组,变量列表)

【例 22】　线性方程组与系数矩阵的转换。

```
> with(LinearAlgebra):
> GenerateEquations(<<1|2>,<3|4>>,[x,y],<5|6>);
```

$$[x+2y=5,3x+4y=6]$$

```
> GenerateMatrix([2*y=5-x,4*y+3*x-6],[x,y]);
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix},\begin{bmatrix} 5 \\ 6 \end{bmatrix}$$

4.5.2 求解线性方程组

假设方程组具有形式 $AX=B$。

BackwardSubstitute(A,B)
ForwardSubstitute(A,B)
NullSpace(A)
LinearSolve(A,B,选项)

当 A 是一个下三角阶梯形矩阵的时候，ForwardSubstitute 利用向前代入的方法依次解得 $x_1,\cdots,x_n$；当 A 是一个上三角阶梯形矩阵的时候，BackwardSubstitute 利用向后代入的方法依次解得 $x_n,\cdots,x_1$；当 $B=0$ 的时候，NullSpace 给出方程的基础解系。一般地，LinearSolve 会自动选择合适的算法求解线性方程组；用户也可以指定算法，如 none，solve，subs，Cholesky，LU，QR，hybrid，SparseLU，SparseDirect，SparseIterative 等。以上函数都是 LinearAlgebra 函数包中的函数。

【例 23】 求解方程组 $\begin{cases}x_1+3x_2+6x_3+2x_4=-1\\6x_1+3x_2+6x_3+12x_4=24\\3x_1-x_2-2x_3+6x_4=17\end{cases}$。

```
> with(LinearAlgebra);
> A,B:=<<1|3|6|2>,<6|3|6|12>,<3|-1|-2|6>>,<-1,24,17>;
```

$$A,B:=\begin{bmatrix}1&3&6&2\\6&3&6&12\\3&-1&-2&6\end{bmatrix},\begin{bmatrix}-1\\24\\17\end{bmatrix}$$

```
> LinearSolve(A,B,free=x); # 定义自由变量 x
```

$$\begin{bmatrix}5-2x_4\\-2-2x_3\\x_3\\x_4\end{bmatrix}$$

```
> ReducedRowEchelonForm(<A|b>);
```

$$\begin{bmatrix}1&0&0&2&5\\0&1&2&0&-2\\0&0&0&0&0\end{bmatrix}$$

```
> U:= GaussianElimination(<A|B>);
```

$$U := \begin{bmatrix} 1 & 3 & 6 & 2 & -1 \\ 0 & -15 & -30 & 0 & 30 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

```
> BackwardSubstitute(U[1..2,1..4],U[1..2,5],free=y);
```

$$\begin{bmatrix} 5-2y_1 \\ -2-2y_2 \\ y_2 \\ y_1 \end{bmatrix}$$

4.5.3 最小二乘解

LeastSquares(A,B)

给出方程组 $AX = B$ 的近似解 X 使得 Norm(AX − B, Frobenius)最小。LeastSquares 是 LinearAlgebra 函数包中的函数。

【例 24】 求方程组 $\begin{cases} x_1+2x_2=5 \\ 2x_1+x_2=6 \\ x_1+x_2=4 \end{cases}$ 的最小二乘解。

```
> A,B:=<<1,2,1>|<2,1,1>>,<5,6,4>;
```

$$A,B := \begin{bmatrix} 1 & 2 \\ 2 & 1 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 5 \\ 6 \\ 4 \end{bmatrix}$$

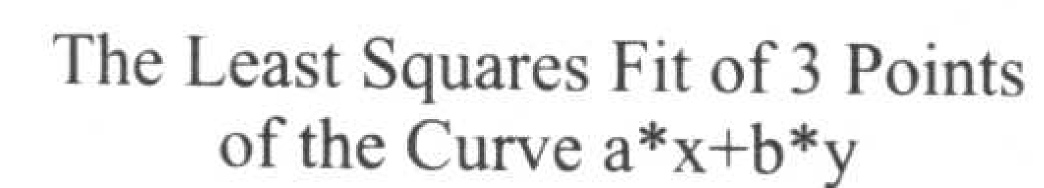
```
> with(LinearAlgebra): X:=LeastSquares(A,B);
```

$$X := \begin{bmatrix} \frac{26}{11} \\ \frac{15}{11} \end{bmatrix}$$

```
> Norm(A.X-b,Frobenius); # 计算误差
```

$$\frac{1}{11}\sqrt{11}$$

```
> unwith(LinearAlgebra);
> with(Student[LinearAlgebra]):
> LeastSquaresPlot([1,2,1],[2,1,1],[5,6,4],[x,y],curve=a*x+b*y);
```



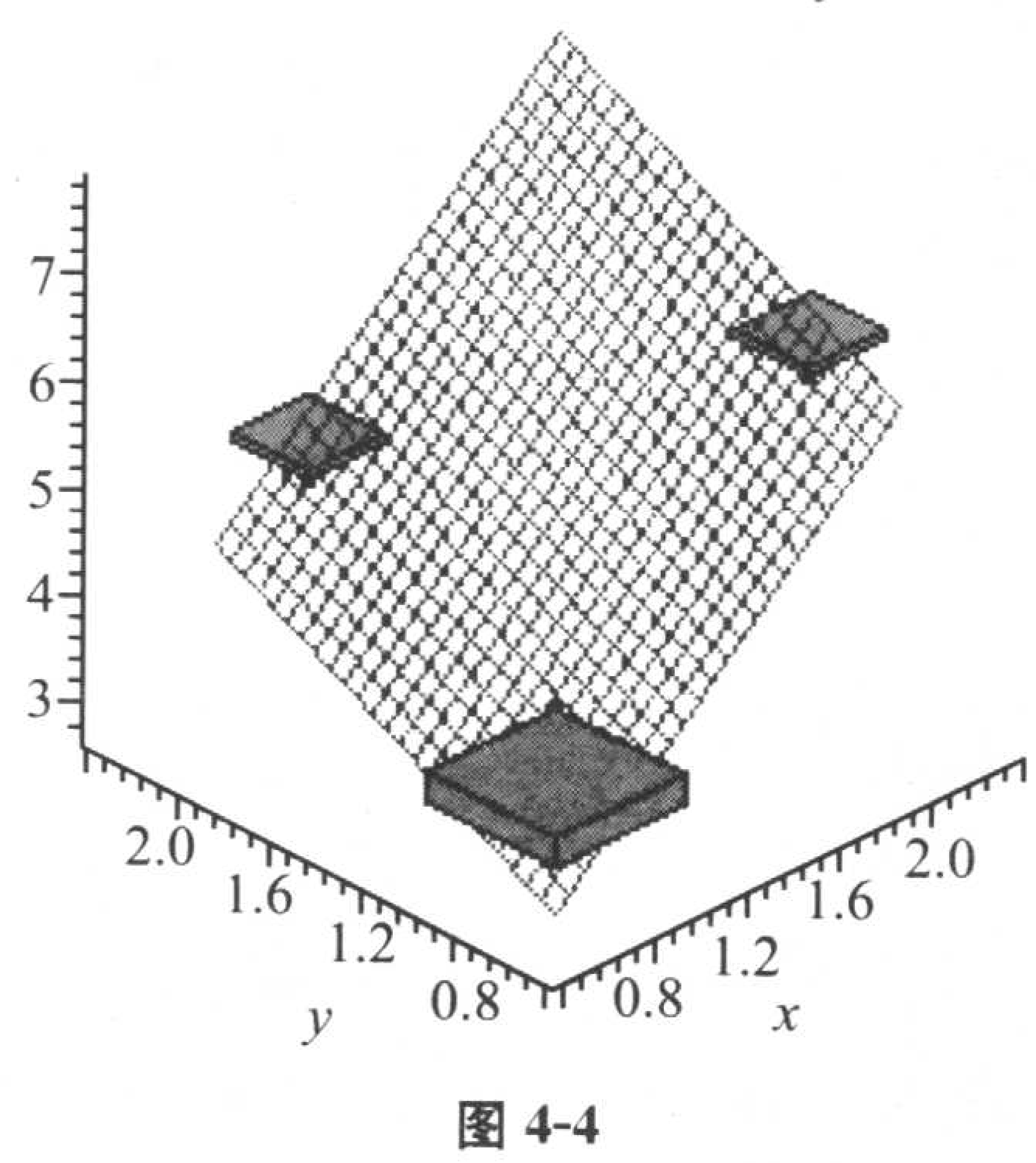


图 4-4

最小二乘解的几何含意是求向量在子空间上的投影。

```
> ProjectionPlot(<5,6,4>,[<1,2,1>,<2,1,1>]);
```

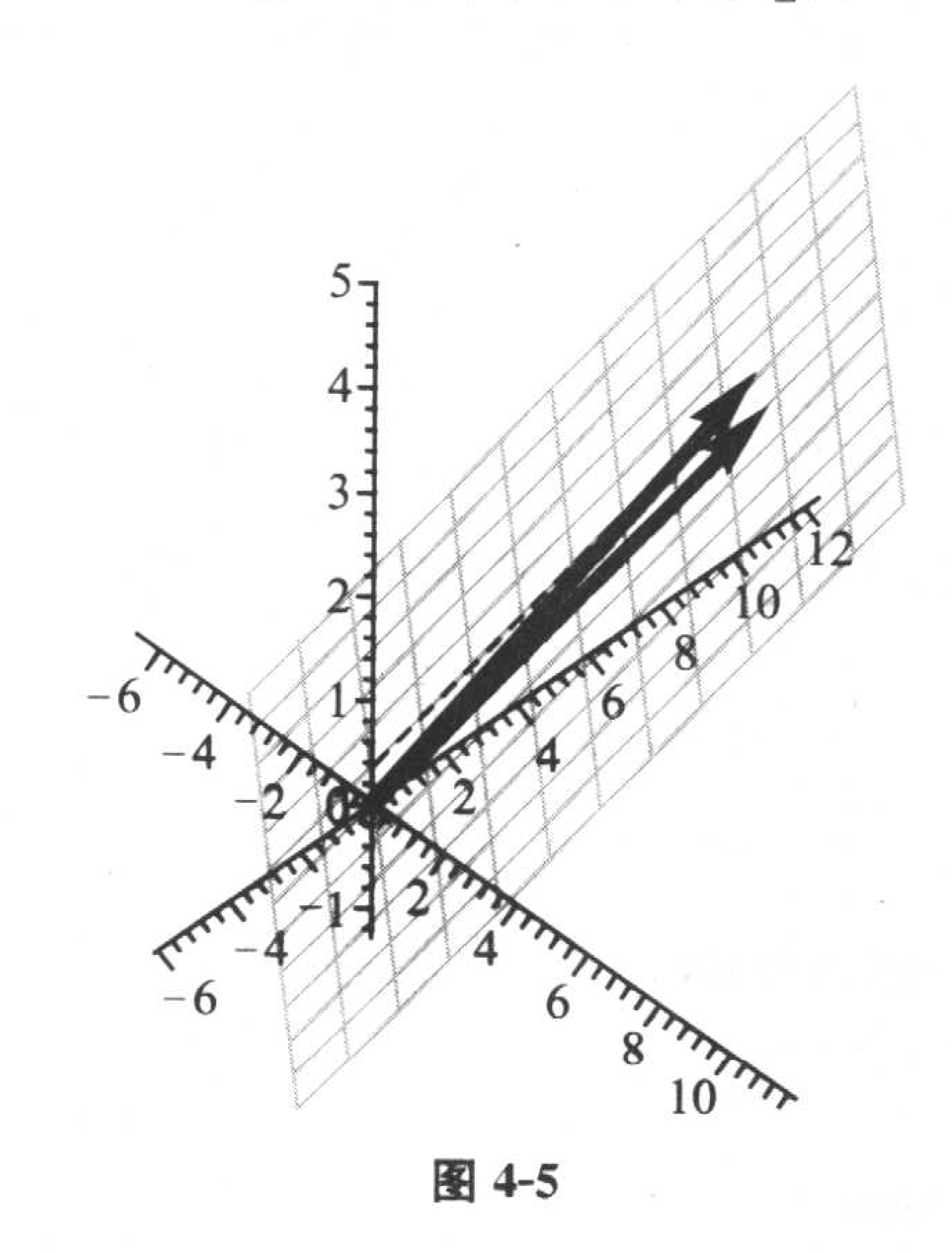


图 4-5

4.6 应用实例

【例 25】（比赛排名）设有 6 个队进行单循环比赛，排名由各队的积分决定：胜队得 1 分，负队得 0 分，平局各得 0.5 分；若两队积分相同时，则比较对手分（战胜的对手的积分之和）；若对手分也相同时，则比较对手的对手分。依次类推下去。已知比赛结果为 1 队负 2 队、1 队胜 3 队、1 队胜 4 队、1 队平 5 队、1 队胜 6 队、2 队胜 3 队、2 队胜 4 队、2 队负 5 队、2 队负 6 队、3 队胜 4 队、3 队胜 5 队、3 队胜 6 队、4 队负 5 队、4 队平 6 队、5 队胜 6 队。请为其排名。

我们用矩阵 $A=(a_{i,j})$ 来记录胜负关系，$a_{i,j}=0$ 表示 i 负 j，$a_{i,j}=\frac{1}{2}$ 表示 i 平 j，$a_{i,j}=1$ 表示 i 胜 j。于是，A 的行和即为各队积分，A^2 的行和即为各队的对手分，A^3 的行和即为各队的对手的对手分……

```
> A := <<0|0|1|1|1/2|1>,<1|0|1|1|0|0>,<0|0|0|1|1|1>,<0|0|0|0|0|1/2>,
  <1/2|1|0|1|0|1/2>,<0|1|0|1/2|1/2|0>>;
  b := <1,1,1,1,1,1>;
  <A.b|A.A.b>;
```

$$\begin{bmatrix} \frac{7}{2} & 7 \\ 3 & 7 \\ 3 & \frac{11}{2} \\ \frac{1}{2} & 1 \\ 3 & \frac{25}{4} \\ 2 & \frac{19}{4} \end{bmatrix}$$

通过比较积分和对手分，排名如下：1、2、3、5、6、4。

【例 26】以图 4-6 为例，求图上任意两个顶点之间的距离，即连接两点的最少边数。

我们用 0－1 矩阵 $A=(a_{i,j})$ 来表示一个图，$a_{i,j}=0$ 表示 i 与 j 不相邻，$a_{i,j}=1$ 表示 i 与 j 相邻；用矩阵 $D=(d_{i,j})$ 来记录顶点间的距离。注意到 $d_{i,j}=\min_k(d_{i,k}+d_{k,j})$。

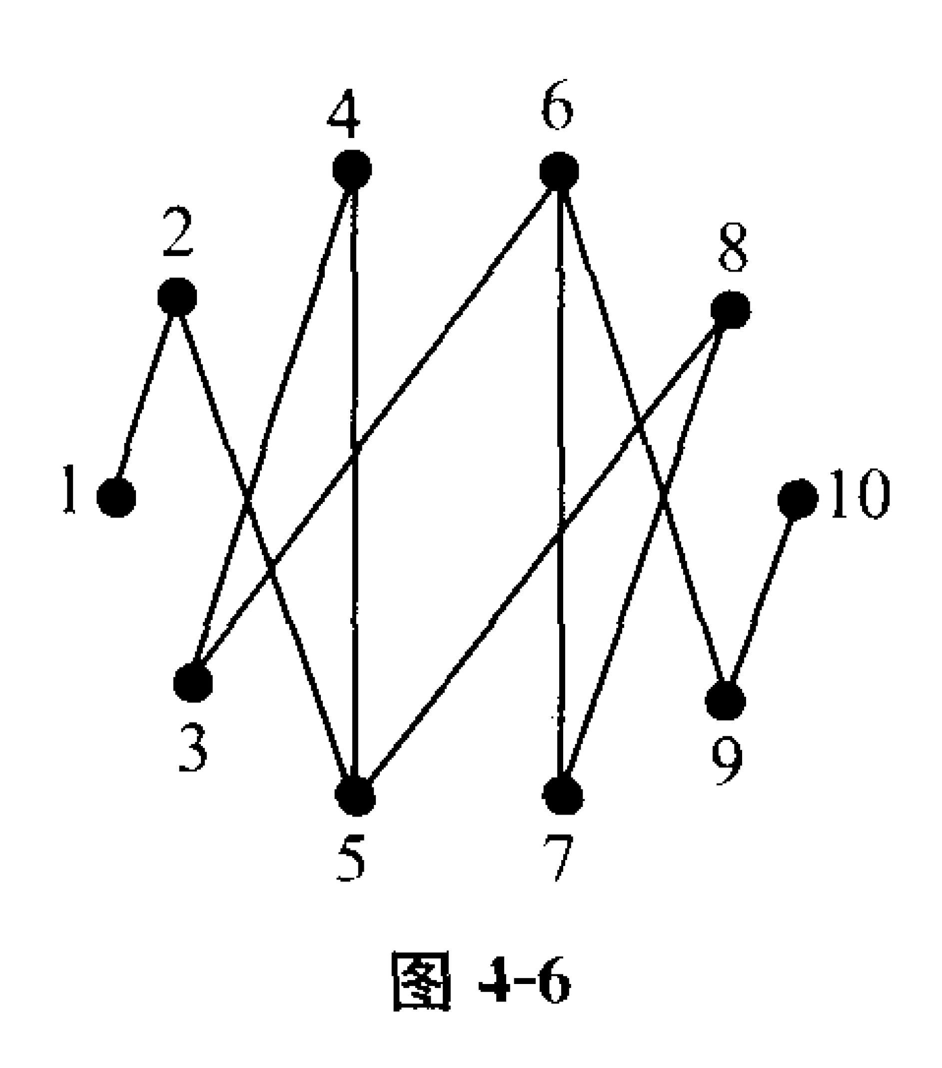


图 4-6

```
> DistanceMatrix := proc(a::Matrix)
  local d,i,j,k,m,n,s;
  uses LinearAlgebra;
  d :=a; m :=1; n :=RowDimension(a);
  while m<n do
    m :=2 * m;
    for i from 2 to n do for j to i-1 do for k to n do
      if d[i,k]>0 and d[k,j]>0 then
        s :=d[i,k]+d[k,j];
        d[i,j] :=`if(d[i,j]=0,s,min(d[i,j],s));
        d[j,i] :=d[i,j]
      end if
     end do end do end do
  end do;
  return d
  end proc;
> A :=Matrix(10); A[1,2] :=1; A[2,5] :=1; A[3,4] :=1; A[3,6] :
  =1;
  A[4,5] :=1; A[5,8] :=1; A[6,7] :=1; A[6,9] :=1; A[7,8] :=
  1;
  A[9,10] :=1; A :=A+A^%T; # 手工输入图的邻接矩阵 A
> DistanceMatrix[A];
```

$$\begin{bmatrix} 0 & 1 & 4 & 3 & 2 & 5 & 4 & 3 & 6 & 7 \\ 1 & 0 & 3 & 2 & 1 & 4 & 3 & 2 & 5 & 6 \\ 4 & 3 & 0 & 1 & 2 & 1 & 2 & 3 & 2 & 3 \\ 3 & 2 & 1 & 0 & 1 & 2 & 3 & 2 & 3 & 4 \\ 2 & 1 & 2 & 1 & 0 & 3 & 2 & 1 & 4 & 5 \\ 5 & 4 & 1 & 2 & 3 & 0 & 1 & 2 & 1 & 2 \\ 4 & 3 & 2 & 3 & 2 & 1 & 0 & 1 & 2 & 3 \\ 3 & 2 & 3 & 2 & 1 & 2 & 1 & 0 & 3 & 4 \\ 6 & 5 & 2 & 3 & 4 & 1 & 2 & 3 & 0 & 1 \\ 7 & 6 & 3 & 4 & 5 & 2 & 3 & 4 & 1 & 0 \end{bmatrix}$$

习　题

1. 计算行列式。

(1) $\begin{vmatrix} 3 & 2 & 1 & 0 \\ 2 & 3 & 2 & 1 \\ 1 & 2 & 3 & 2 \\ 0 & 1 & 2 & 3 \end{vmatrix}$　　(2) $\begin{vmatrix} a & b & c & d \\ -b & a & d & -c \\ -c & -d & a & b \\ -d & c & -b & a \end{vmatrix}$

2. 求多项式 $f(x)=\begin{vmatrix} 2x & x & 1 & 2 \\ 1 & x & -2 & -1 \\ 3 & 2 & x & x \\ 1 & 1 & 0 & x \end{vmatrix}$ 中 x^3 和 x^4 项的系数。

3. 设 $A=\begin{pmatrix} 1 & 3 & -1 \\ 6 & -2 & 3 \end{pmatrix}$，$B=\begin{pmatrix} 2 & -1 & -4 \\ 3 & 0 & -2 \\ 0 & 3 & 2 \end{pmatrix}$，$C=\begin{pmatrix} 0 & -2 \\ -1 & 2 \\ 3 & 1 \end{pmatrix}$，计算 AB，AC，B^2，CA 和 ABC。

4. 判断向量组是否线性相关？

(1) $a_1=\begin{pmatrix} 1 \\ -2 \\ 3 \end{pmatrix}$，$a_2=\begin{pmatrix} -2 \\ 3 \\ -1 \end{pmatrix}$，$a_3=\begin{pmatrix} 1 \\ -1 \\ 3 \end{pmatrix}$

(2) $a_1=\begin{pmatrix}2\\-1\\3\\5\end{pmatrix}, a_2=\begin{pmatrix}1\\-1\\1\\2\end{pmatrix}, a_3=\begin{pmatrix}-5\\3\\1\\1\end{pmatrix}, a_4=\begin{pmatrix}3\\3\\2\\0\end{pmatrix}$

5. 求矩阵的秩。

(1) $A=\begin{pmatrix}2 & 1 & -1 & 1 & 1\\3 & -2 & 1 & -3 & 4\\2 & -6 & 4 & 5 & -2\end{pmatrix}$　　(2) $B=\begin{pmatrix}1 & 3 & 5 & -1 & 2\\2 & -1 & -3 & 4 & 3\\5 & 1 & -1 & 7 & 2\\1 & -1 & 2 & -5 & 3\end{pmatrix}$

6. 求矩阵的逆矩阵。

(1) $A=\begin{pmatrix}-2 & 1 & 3\\0 & 3 & 1\\1 & 2 & 0\end{pmatrix}$　　(2) $B=\begin{pmatrix}3 & 3 & -4 & -3\\0 & 6 & 1 & -1\\-3 & 4 & 2 & 1\\2 & 3 & -3 & 2\end{pmatrix}$

7. 解线性方程组。

(1) $\begin{pmatrix}1 & -2 & 1 & 1\\1 & -2 & 1 & -1\\1 & -2 & 1 & 5\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix}=\begin{pmatrix}1\\2\\5\end{pmatrix}$

(2) $\begin{pmatrix}1 & -2 & -1 & -2\\4 & 1 & 3 & 1\\2 & 5 & 4 & -1\\1 & 2 & 1 & 1\end{pmatrix}\begin{pmatrix}x_1\\x_2\\x_3\\x_4\end{pmatrix}=\begin{pmatrix}2\\3\\0\\1\end{pmatrix}$

8. 设线性变换 A 在基 $a_1=(1,-1)$，$a_2=(1,1)$ 下的矩阵是 $\begin{bmatrix}2 & 3\\-2 & 1\end{bmatrix}$，求 A 在基 $b_1=(2,0)$，$b_2=(-1,1)$ 下的矩阵。

9. 求矩阵的全部特征值和特征向量。

(1) $A=\begin{pmatrix}0 & 2a\\-a & 0\end{pmatrix}$　　(2) $B=\begin{pmatrix}0 & 0 & 3\\0 & 2 & 0\\1 & 0 & 0\end{pmatrix}$

(3) $C=\begin{pmatrix}1 & 1 & 1 & 1\\1 & 1 & -1 & -1\\1 & -1 & 1 & -1\\1 & 1 & -1 & 1\end{pmatrix}$

10. 判断二次型是否为正定二次型。

(1) $Q(x_1,x_2,x_3)=x_1^2-2x_2^2+x_3^2+2x_1x_2+4x_1x_3+2x_2x_3$

(2) $Q(x_1,x_2,x_3,x_4)=x_1x_2+x_2x_3+x_3x_4+x_4x_1$

11. 用正交变换化二次型为标准形。

(1) $Q=2x_1^2+x_2^2-4x_1x_2-4x_2x_3$

(2) $Q=3x_1^2+4x_1x_2+8x_1x_3+4x_2x_3+3x_3^2$

12. 计算下列矩阵的 Jordan 标准形并写出过渡矩阵。

(1) $A=\begin{pmatrix} 3 & 7 & -3 \\ -2 & -5 & 2 \\ -4 & -10 & -3 \end{pmatrix}$

(2) $B=\begin{pmatrix} 3 & -4 & 0 & 2 \\ 4 & -5 & -2 & 4 \\ 0 & 0 & 3 & -2 \\ 0 & 0 & 2 & 1 \end{pmatrix}$

第5章　微分方程

5.1　常微分方程

5.1.1　常微分方程的求解和作图命令

1. 求解微分方程命令 dsolve

在微分方程中，我们称只有一个自变量的微分方程为常微分方程，具有两个或两个以上自变量的微分方程为偏微分方程。例如：描述物体冷却过程的数学模型

$$\frac{\mathrm{d}u}{\mathrm{d}t}=-k(u-u_0)$$

含有自变量 t、未知函数 $u(t)$ 以及一阶导数 $\frac{\mathrm{d}u}{\mathrm{d}t}$，是一个常微分方程。Maple 用 dsolve 求解常微分方程或常微分方程组，其用法有：

```
dsolve(常微方程)
dsolve(常微方程,待解函数,选项)
dsolve({常微方程,初值},待解函数,选项)
dsolve({常微方程组,初值},{待解函数},选项)
```

其中选项设置解的求解方法和解的表示方式。求解方法有 type＝formal_series（形式幂级数解）、type＝formal_solution（形式解）、type＝numeric（数值解）、type＝series（级数解）、method＝fourier（通过 Fourier 变换求解）、method＝laplace（通过 Laplace 变换求解）等。解的表示方式有 explicit（显式）、implicit（隐式）、parametric（参数式）。当方程比较复杂时，要想得到显式解通常会十分困难，结果也会相当复杂。这时，方程的隐式解更为有用，一般也要简单得多。dsolve 为标准库函数。

2. 方程数值解作图命令 odeplot

要作出常微分方程数值解的图像，请使用 odeplot 函数。odeplot 在函数包 plots 中，可通过 with(plots)或 plots[odeplot]调出。

odeplot(数值解，被绘函数，参数范围，选项)

5.1.2 一阶常微分方程

1. 可分离变量方程

若一阶微分方程具有形式 $\frac{dy}{dx}=f(x)g(y)$，则称为可分离变量方程。一般可以通过对方程$\frac{dy}{g(y)}=f(x)dx$ 两边分别积分，得到方程的隐式解。

【例 1】 求解微分方程 $y'(x)=\frac{\sin(x)}{\sin(y)}$。

```
> eq: =diff(y(x),x)=sin(x)/sin(y(x));
```

$$eq:=\frac{d}{dx}y(x)=\frac{\sin(x)}{\sin(y(x))}$$

显然，这是可分离变量的常微分方程。用 Detools 函数包中的 odeadvisor 函数检测方程的类型，输出结果_separable 说明方程 *eq* 是可分离变量类型。

```
> DEtools[odeadvisor](eq);
```

[_separable]

用 dsolve 函数求解方程，得到方程的通解。

```
> dsolve(eq);
```

$$y(x)=\pi-\arccos(-\cos(x)+_C1)$$

设定选项 implicit，得到方程的隐式解。

```
> dsolve(eq,implicit);
```

$$-\cos(x)+\cos(y(x))+_C1=0$$

附加初始值 $y(0)=1$，得到方程的准确解。

```
> dsolve({eq,y(0)=1});
```

$$y(x)=\arccos(\cos(x)-1+\cos(1))$$

调用 odeplot 命令作出方程解的图形(图 5-1)。

```
> dsolve({eq,y(0)=1},numeric,range=-2..2);
```

proc(*x_rkf*45)…end proc

```
> plots[odeplot](%);
```

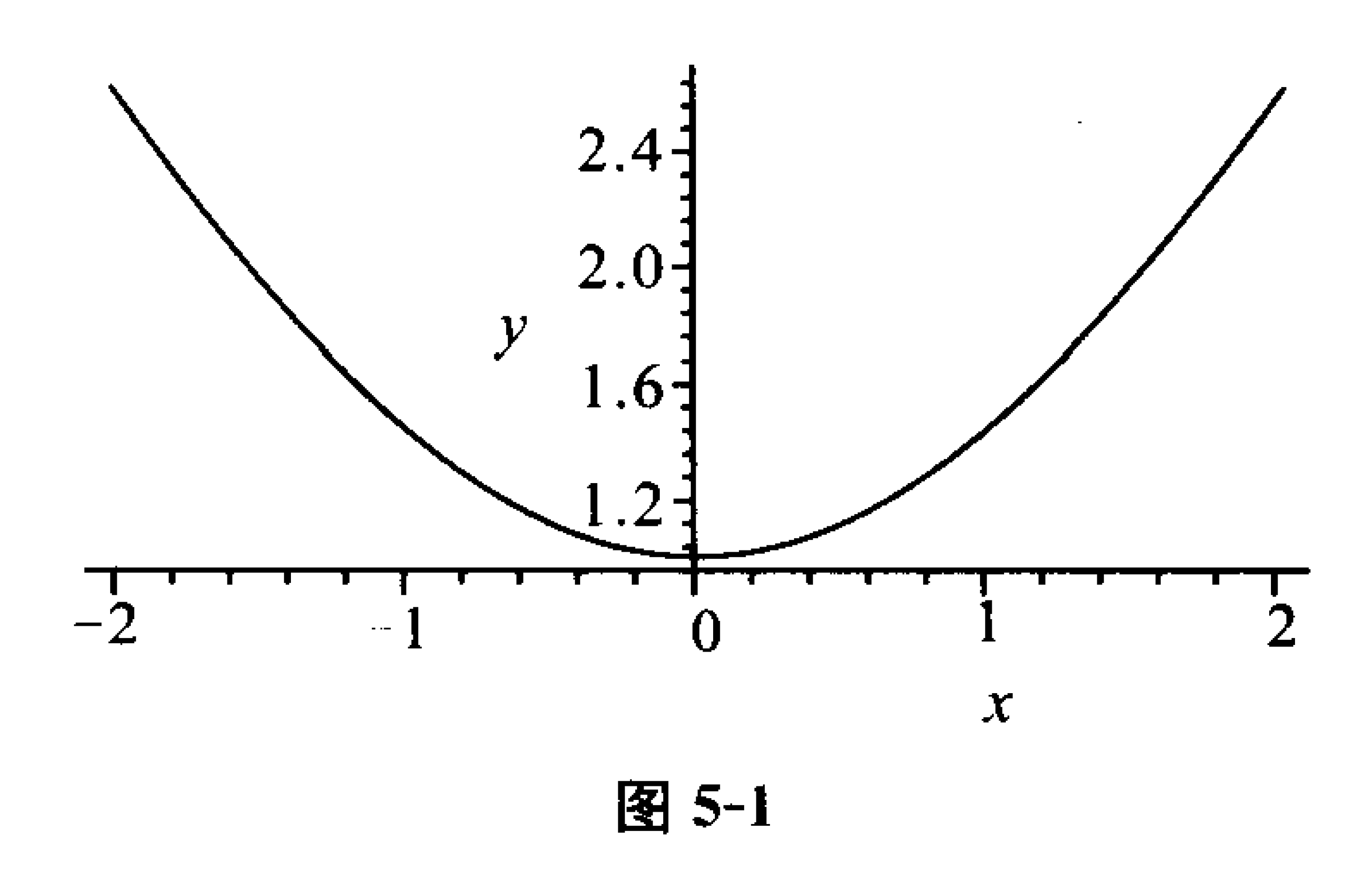


图 5-1

2. 齐次方程

若常微分方程具有形式 $\frac{\mathrm{d}y}{\mathrm{d}x}=f\left(\frac{y}{x}\right)$，则称为齐次方程。通常用变量代换 $u=\frac{y}{x}$ 将齐次方程转换为可分离变量方程 $\frac{\mathrm{d}u}{\mathrm{d}x}=\frac{f(u)-u}{x}$。

【例 2】 求解齐次微分方程 $\frac{\mathrm{d}y}{\mathrm{d}x}=\frac{y}{x}+\tan\left(\frac{y}{x}\right)$。

```
> eq := D(y)(x)=y(x)/x+tan(y(x)/x);
```

$$eq := (\mathrm{D}(y))(x)=\frac{y(x)}{x}+\tan\left(\frac{y(x)}{x}\right)$$

调用 odeadvisor 可以看到方程的类型，_homogeneous 表明 *eq* 是齐次方程。

```
> DEtools[odeadvisor](eq);
```

[[_homogeneous, class *A*], _dAlembert]

```
> dsolve(eq); # 直接求解得到方程的显式解
```

$$y(x)=\arcsin(x_C1)x$$

```
> dsolve({eq,y(1)=1});
```

$$y(x)=x\arcsin(x\sin(1))$$

```
> dsolve(eq,implicit); # 求方程的隐式解
```

$$\ln\left(\sin\left(\frac{y(x)}{x}\right)\right)-\ln(x)-_C1=0$$

```
> dsolve({eq,y(1)=3},implicit);
```

$$\ln\left(\sin\left(\frac{y(x)}{x}\right)\right)-\ln(x)-\ln(\sin(3))=0$$

```
> dsolve({eq,y(1)=3},numeric,range=1..6); # 取初值 y(1)=3, 1<x<6
```

proc(*x_rkf45*)…end proc



> plots[odeplot](%)；# 作出方程解的图形

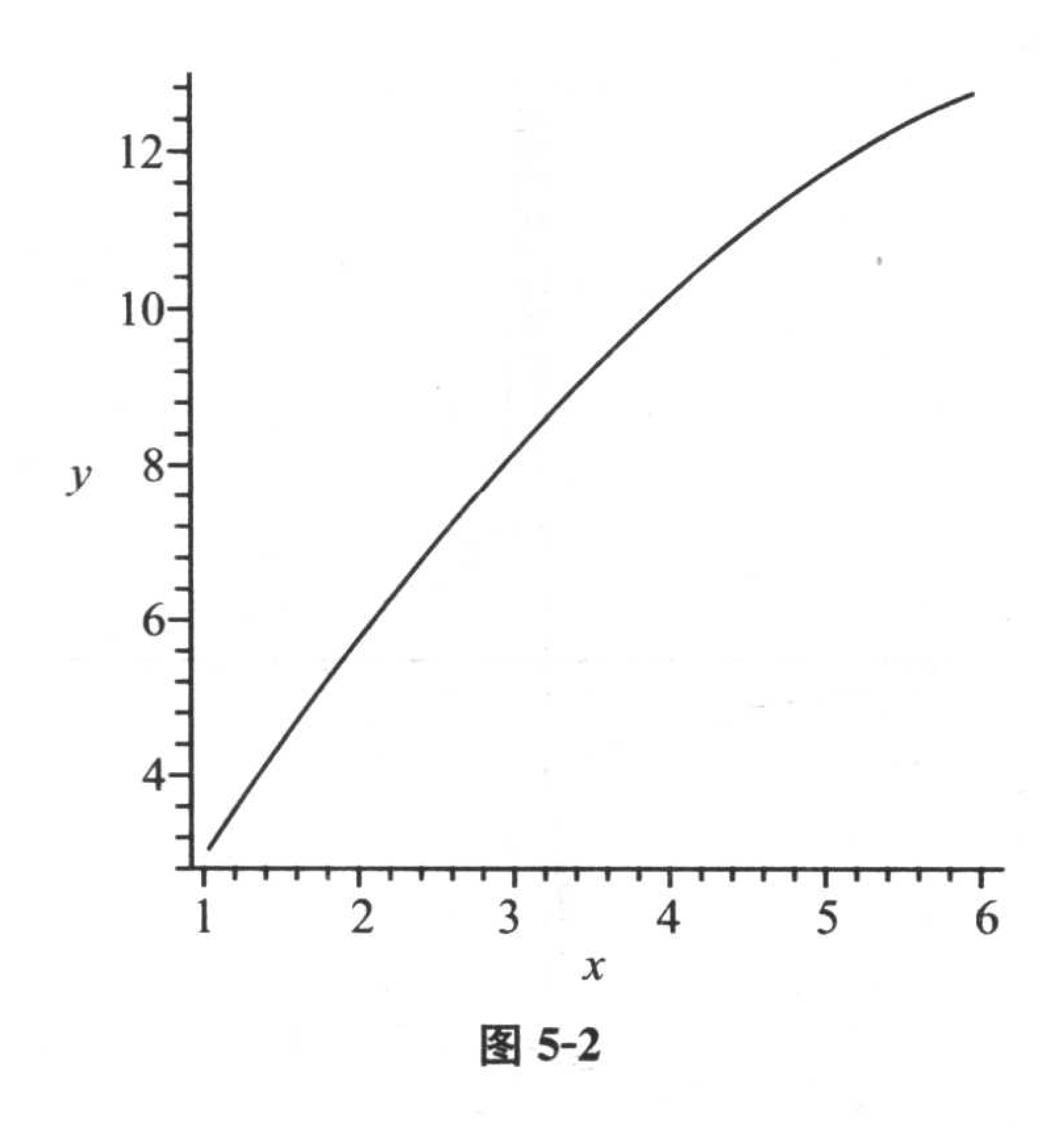


图 5-2

3. 线性方程

形如 $y'(x)+p(x)y=q(x)$ 的微分方程称为一阶线性方程，在 Maple 中既可以用 dsolve 函数求解，也可以用 DEtools 函数包中的 linearsol 函数求解。linearsol 是专门用于求解线性微分方程的命令，使用格式为：

linearsol(线性方程，待解函数)

linearsol 的返回值为集合形式的解。

【例 3】 求解线性方程 $\dfrac{\mathrm{d}y}{\mathrm{d}x}=\dfrac{\sin(x)-y}{x}$。

> eq：=D(y)(x)=(sin(x)－y(x))/x；

$$eq := (\mathrm{D}(y))(x)=\frac{\sin(x)-y(x)}{x}$$

> DEtools[sodeadvisor](eq)；# _linear 表明方程为线性方程

[_linear]

> dsolve(eq)；

$$y(x)=\frac{-\cos(x)+_C1}{x}$$

> DEtools[linearsol](eq)；# 请比较两个解函数形式

$$\left\{y(x)=\frac{-\cos(x)+_C1}{x}\right\}$$

> dsolve({eq,y(1)=2},numeric,range=－5..5)：# 取初值 y(1)=2，－5<x<5

> plots[odeplot](%)； # 作出解的图像

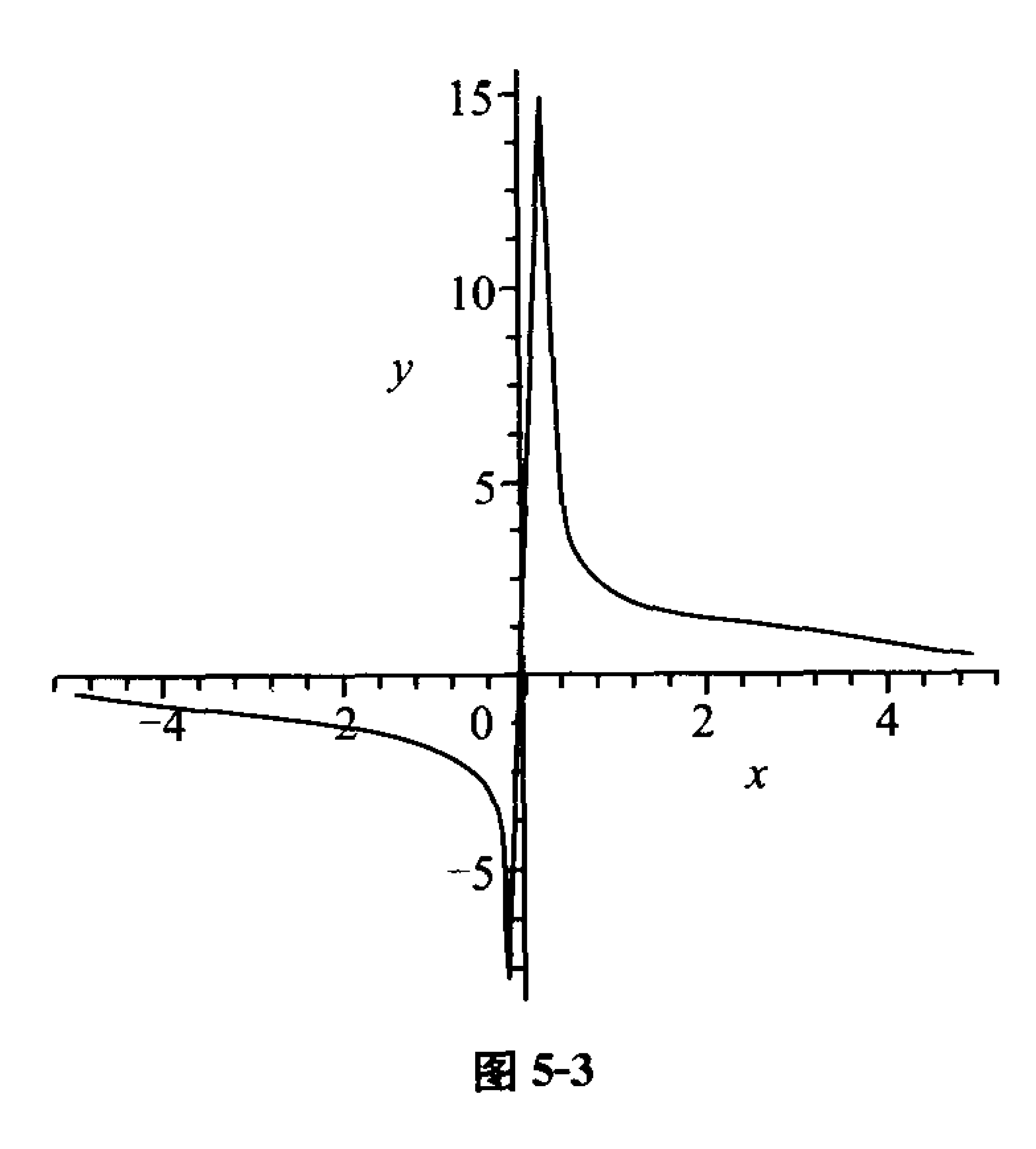


图 5-3

4. Bernoulli 方程

形如 $\frac{\mathrm{d}y}{\mathrm{d}x}+p(x)y=q(x)y^n$ 的微分方程称为 Bernoulli 方程。当 $n\neq 0,1$ 时，变量代换 $u=y^{1-n}$ 将 Bernoulli 方程转化为线性方程 $\frac{\mathrm{d}u}{\mathrm{d}x}=(n-1)(p(x)\cdot u-q(x))$。

【例 4】 求解 Bernoulli 方程 $\frac{\mathrm{d}y}{\mathrm{d}x}=6x^{-1}y-xy^2$。

> eqa：=D(y)(x)=6 * y(x)/x−x * y(x)^2;

$$eqa := \mathrm{D}(y)(x) = \frac{6y(x)}{x} - xy(x)^2$$

调用 odeadvisor 查看方程的类型，表明这是个 Bernoulli 方程，并具有齐次和有理属性。

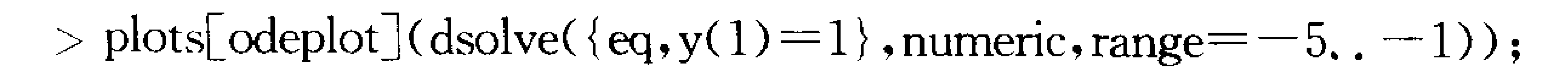
> DEtools[odeadvisor](eqa)；

[[_homogeneous, class G], _rational, _Bernoulli]

> dsolve(eqa)； # 直接求解

$$y(x) = \frac{8x^6}{x^8+8_C1}$$

给定方程一个初值 $y(1)=1$，$-5<x<-1$，作出它的解的图像(图 5-4)。




```
> plots[odeplot](dsolve({eq,y(1)=1},numeric,range=-5..-1));
```

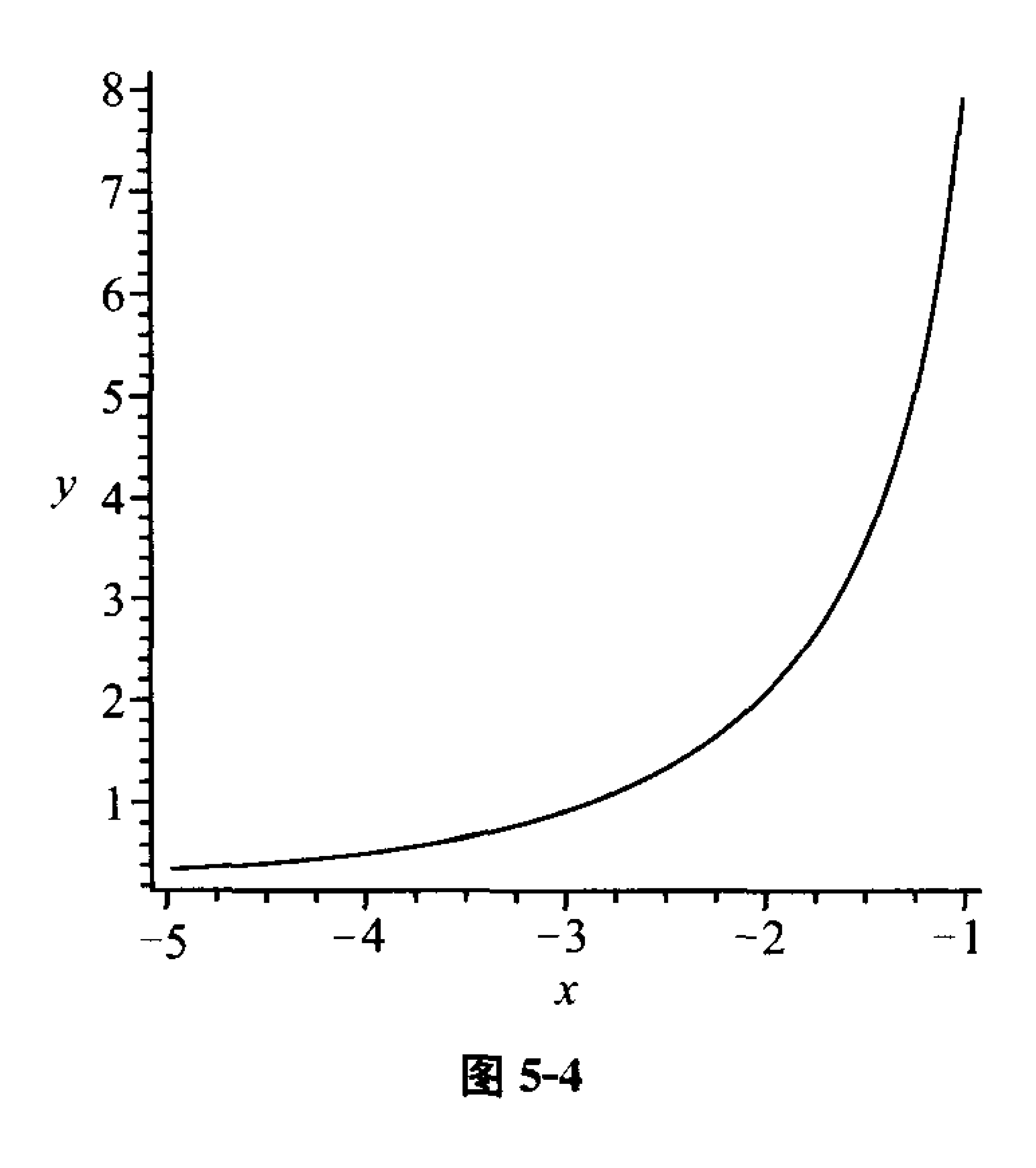


图 5-4

5. Riccati 方程

形如 $\frac{\mathrm{dy}}{\mathrm{d}x}=p(x)y^2+q(x)y+r(x)$ 的微分方程称为 Riccati 方程。在一般情况下，无法求出其通解的解析表达式。但是如果知道一个特解，则可求出它的通解。

【例 5】 求解 Riccati 方程 $\frac{\mathrm{dy}}{\mathrm{d}x}=\frac{1}{x^2}-y^2$。

```
> eq:=D(y)(x)=1/x^2-y(x)^2;
```

$$eq:=(\mathrm{D}(y))(x)=\frac{1}{x^2}-y(x)^2$$

调用 odeadvisor 判别方程的类型，表明这是 Riccati 方程，并具有齐次和有理属性。

```
> DEtools[odeadvisor](eq);
```

[[_homogeneous, class G], _rational, _Riccati]

```
> dsolve(eq,implicit); # 求隐式解
```

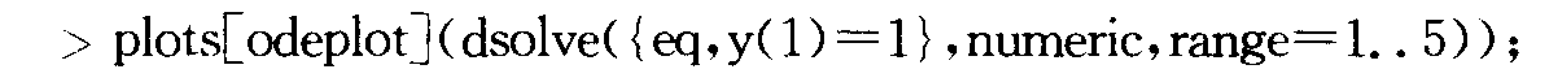
$$\ln(x)-_C1-\frac{2}{5}\sqrt{5}\operatorname{arctanh}\left(\left(\frac{2}{5}y(x)x-\frac{1}{5}\right)\sqrt{5}\right)=0$$

这是原方程的隐式解，我们再给出它的显式解，请观察和比较显式解和隐式解。

```
> dsolve(eq); # 直接求解
```

$$y(x)=\frac{1}{2}\,\frac{1-\sqrt{5}\tanh\left(-\frac{1}{2}\sqrt{5}\ln(x)+\frac{1}{2}\sqrt{5}_C1\right)}{x}$$

然后给定方程一个初值 $y(1)=1$，$1<x<5$，作出它的解的图像(图 5-5)。



```
> plots[odeplot](dsolve({eq,y(1)=1},numeric,range=1..5));
```

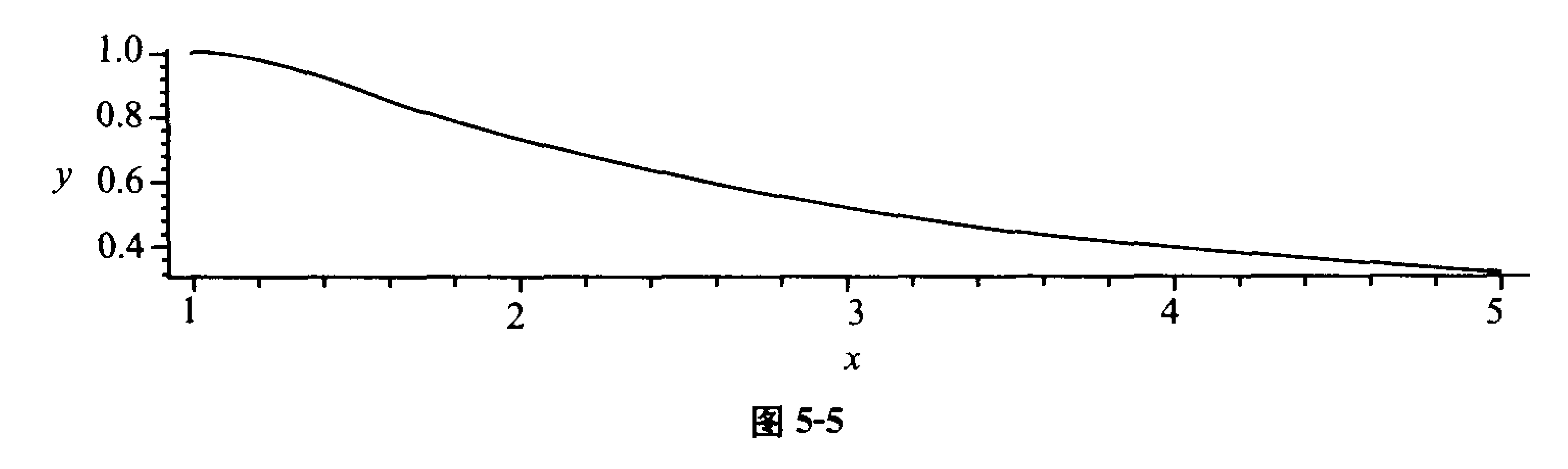


图 5-5

5.1.3 常系数线性微分方程

形如 $y^{(n)}+c_1y^{(n-1)}+\cdots+c_ny=f(x)$，其中 $c_1,\cdots,c_n$ 为常数的微分方程被称为 n 阶常系数线性微分方程。当 $f(x)=0$ 时，称方程为齐次方程；当 $f(x)\neq 0$ 时，称为非齐次方程。

1. 齐次方程

【例 6】 求解 $y''+2y'+y=0$。这是一个二阶常系数齐次微分方程。

```
> eq:=diff(y(x),x$2)+2*diff(y(x),x)+y(x)=0;
```

$$eq := \frac{d^2}{dx^2}y(x)+2\left(\frac{d}{dx}y(x)\right)+y(x)=0$$

```
> dsolve(eq); # 用 dsolve 函数直接求解
```

$$y(x) = _C1e^{(-x)}+_C2e^{(-x)}x$$

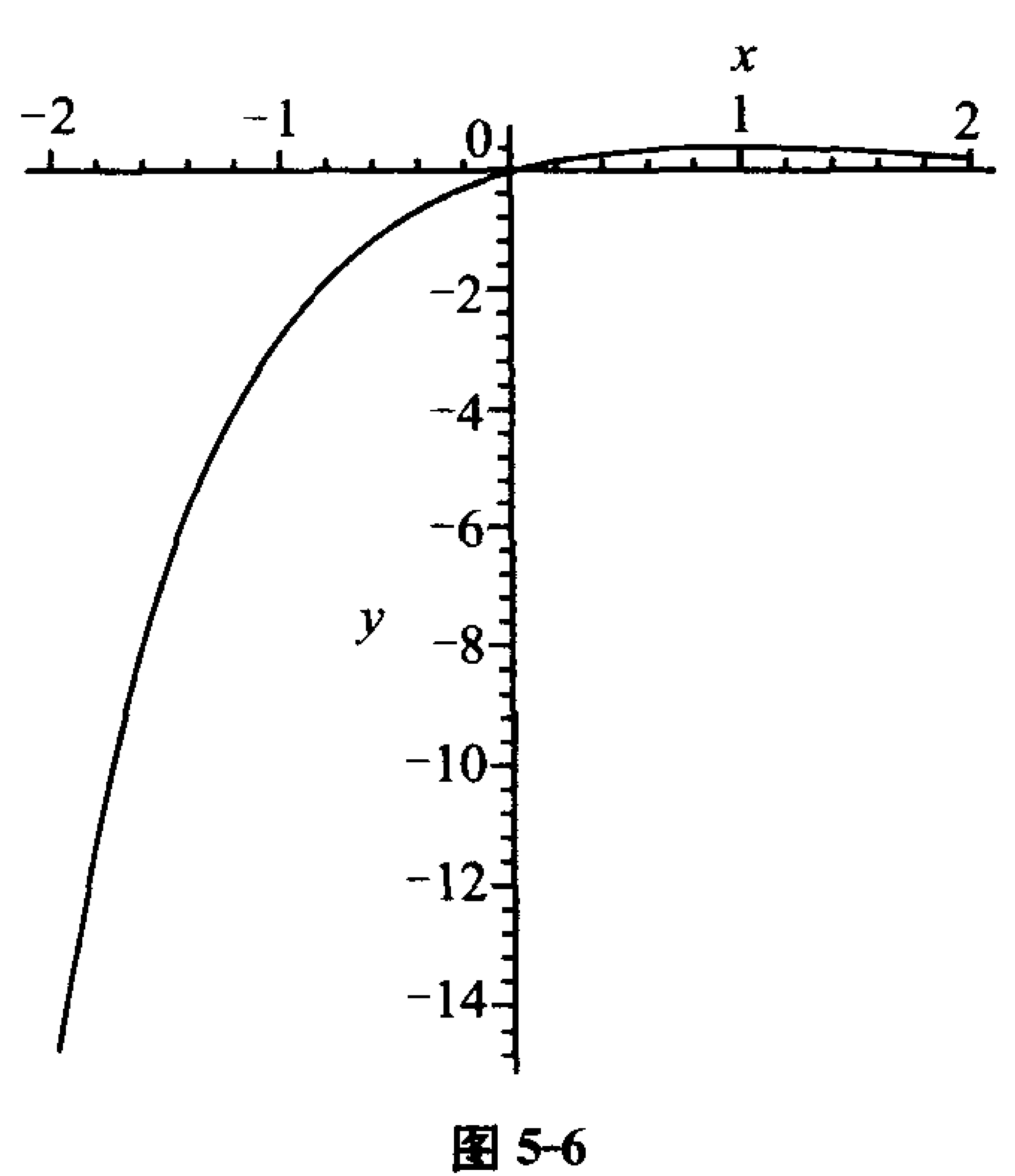


图 5-6

除了上述 dsolve 函数外，还可以调用函数包 DEtools 中的函数 constcoeffsols，求出微分方程的线性无关的特解，这些特解的线性组合就是方程的所有解。

```
> DEtools[constcoeffsols](eq);
```

$$[e^{(-x)}, e^{(-x)}x]$$

给定方程一个初值 $y(0)=0, y'(0)=1, -2<x<2$，作出它的解的图像(图 5-6)。

```
> plots[odeplot](dsolve({eq,y(0)=0,D(y)(0)=1},numeric,range=-2..2));
```

2. 非齐次方程

由于非齐次方程解的具体讨论比较复杂，这里只举例说明用 Maple 求解。想了解更详细的过程，可以参考有关微分方程的书籍。

【例 7】 求解常系数线性非齐次方程 $y''+y'=2x^2-3$。

```
> eq:=diff(y(x),x$2)+diff(y(x),x)=2*x^2-3;
```

$$eq := \frac{d^2}{dx^2}y(x)+\frac{d}{dx}y(x)=2x^2-3$$

对于上述定义的常系数微分方程，我们调用 dsolve 函数直接求其通解，结果如下。

```
> dsolve(eq,y(x));
```

$$y(x)=-2x^2+\frac{2}{3}x^3-e^{(-x)}_C1+x+_C2$$

给定方程一个初值 $y(0)=0, y'(0)=1, -3<x<3$，作出它的解的图像(图 5-7)。

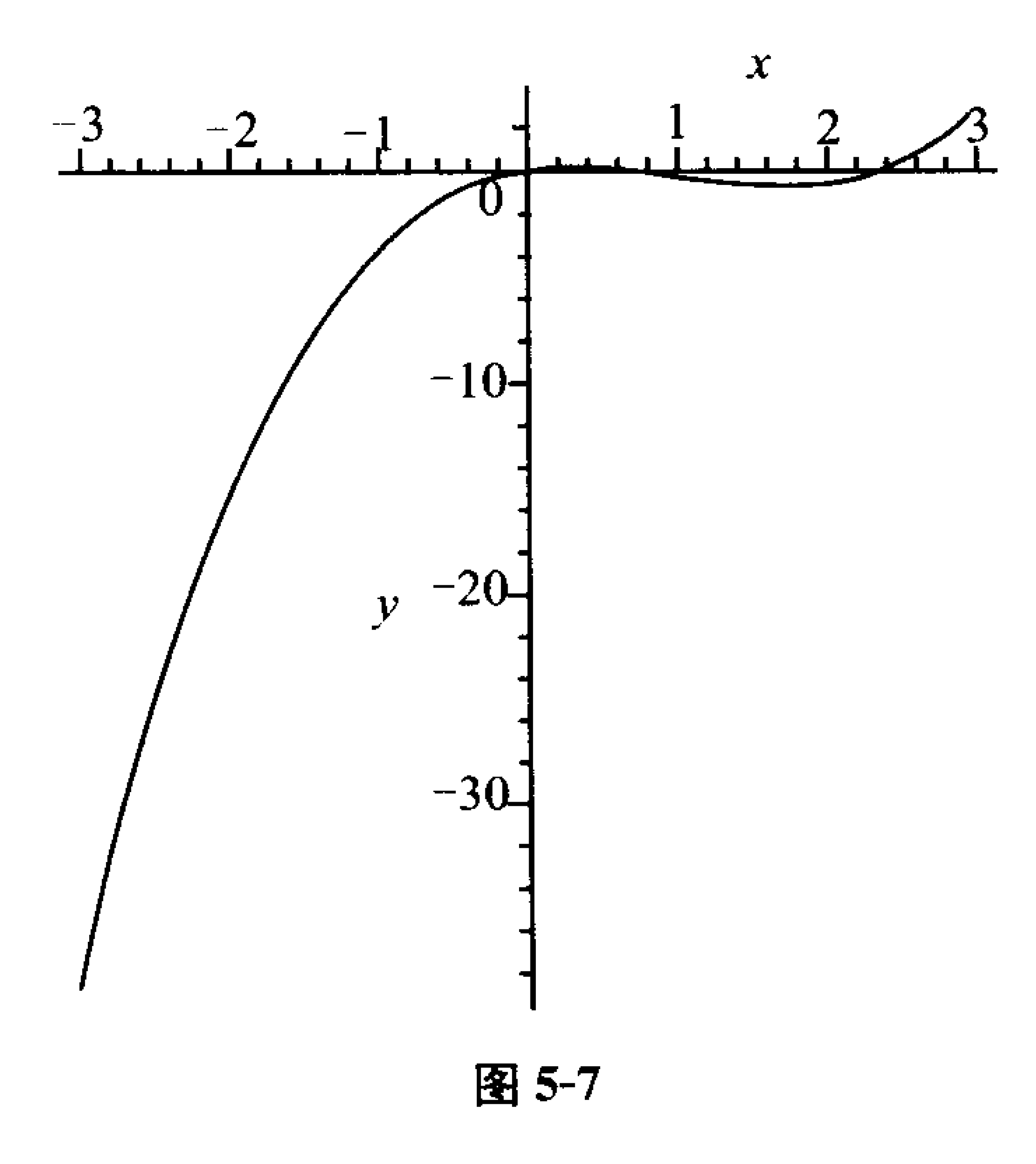


图 5-7

```
> plots[odeplot](dsolve({eq,y(0)=0,D(y)(0)=1},numeric,range=-3..3));
```

3. Euler 方程

齐次 Euler 方程的一般形式为：

$$x^n y^{(n)} + a_1 x^{n-1} y^{(n-1)} + \cdots + a_{n-1} x y' + a_n y = 0$$

其中 $a_1, a_2, \cdots, a_n$ 都是常数。求解方法是做变换 $x = e^t$，求出 $y(t)$，然后再将 $t = \ln x$ 代入，即可以得到原方程的解 $y(x)$。

【例 8】 求解 Euler 方程 $x^2 y'' + 5xy' + 13y = 0$ 。

```
> eq：=x^2 * diff(y(x),x$2)+5 * x * diff(y(x),x)+13 * y(x)=0;
```

$$eq := x^2\left(\frac{d^2}{dx^2}y(x)\right) + 5x\left(\frac{d}{dx}y(x)\right) + 13y(x) = 0$$

```
> DEtools[odeadvisor](eq);
```

$$[[_Emden, _Fowler]]$$

```
> dsolve(eq);
```

$$y(x) = \frac{_C1\sin(3\ln(x))}{x^2} + \frac{_C2\cos(3\ln(x))}{x^2}$$

给定 Euler 方程一个初值 $y(1)=0, y'(1)=1, 1<x<5$，作出它的解的图像（图 5-8）。

```
> plots[odeplot](dsolve({eq,y(1)=0,D(y)(1)=1},numeric,range=1..5));
```

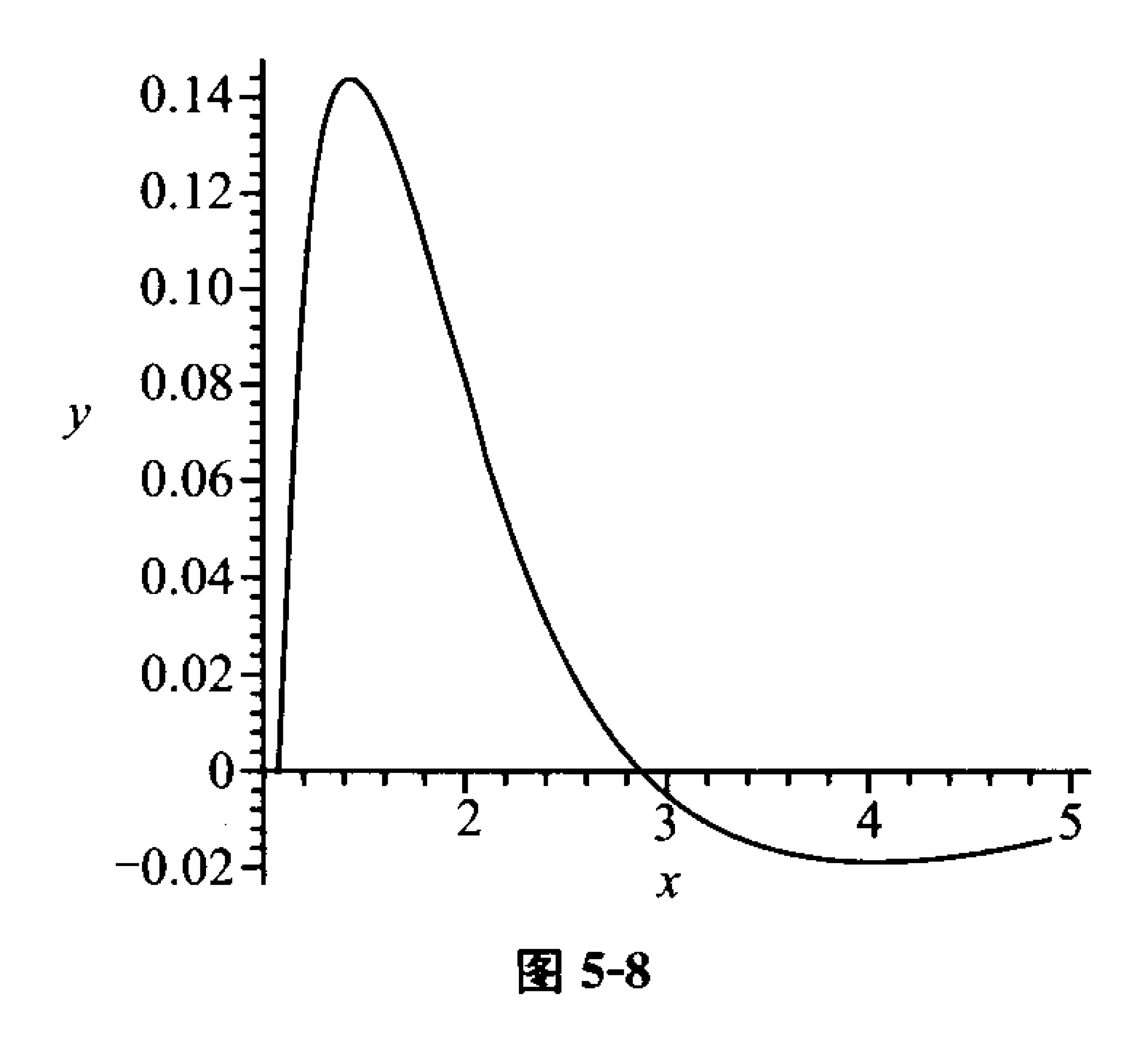


图 5-8

5.1.4 微分方程组

除了单个微分方程之外，dsolve 函数也可以求解微分方程组，求解的方法与 solve 求解方程组类似。

【例 9】 求解线性微分方程组$\begin{cases}x'(t)=2x(t)+y(t)\\ y'(t)=3x(t)+4y(t)\end{cases}$。

```
> eqs：={diff(x(t),t)=2*x(t)+y(t),diff(y(t),t)=3*x(t)+4*y(t)};
```

$$eqs := \left\{\frac{\mathrm{d}}{\mathrm{d}t}x(t) = 2x(t)+y(t), \frac{\mathrm{d}}{\mathrm{d}t}y(t) = 3x(t)+4y(t)\right\}$$

仍然调用 dsolve 函数求解，解函数中的_C1 和_C2 表示任意常数。

```
> dsolve(eqs);
```

$$\{x(t) = _C1\mathrm{e}^{t} + _C2\mathrm{e}^{(5t)}, y(t) = -_C1\mathrm{e}^{t} + 3_C2\mathrm{e}^{(5t)}\}$$

给定方程组一个初值 $x(0)=y(0)=1$，$-1<t<1$，作出它的解的图像(图 5-9)。

```
> dsolve(eqs∪{x(0)=1,y(0)=1},numeric,range=-1..1);
```

proc(x_rkf45)…end proc

```
> plots[odeplot](%,[t,x(t),y(t)]);
```

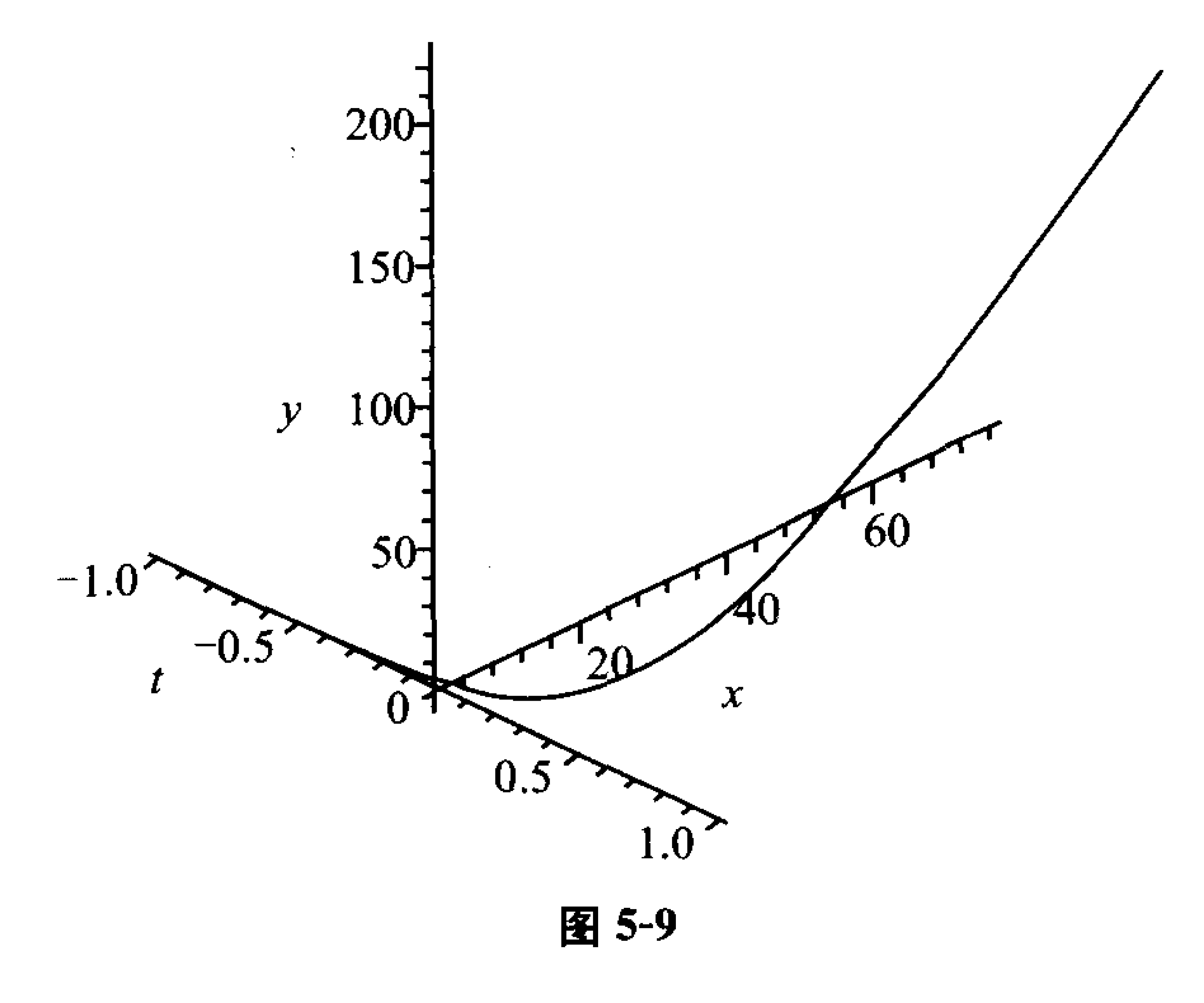


图 5-9

【例 10】 求解非齐次微分方程组$\begin{cases}x'(t)=y(t)-5\cos(t)\\ y'(t)=2x(t)+y(t)\end{cases}$。

虽然是非齐次方程组，仍然用 dsolve 求解，得到方程组的解。

```
> eqs：={diff(x(t),t)=y(t)-5*cos(t),diff(y(t),t)=2*x(t)+y(t)};
```

$$eqs := \left\{\frac{\mathrm{d}}{\mathrm{d}t}x(t) = y(t)-5\cos(t), \frac{\mathrm{d}}{\mathrm{d}t}y(t) = 2x(t)+y(t)\right\}$$

```
> dsolve(eqs);
```

$$\{y(t) = -\mathrm{e}^{(-t)}_C2 + 2\mathrm{e}^{(2t)}_C1 + 3\cos(t) + \sin(t),$$
$$x(t) = \mathrm{e}^{(-t)}_C2 + \mathrm{e}^{(2t)}_C1 - 2\sin(t) - \cos(t)\}$$

给定方程组一个初值 $x(0)=y(0)=1$，$-3<t<3$，作出它的解的图像(图 5-10)。

```
> dsolve(eqs∪{x(0)=1,y(0)=1},numeric,range=-3..3);
```

```
plots[odeplot](%,[t,x(t),y(t)]);
```

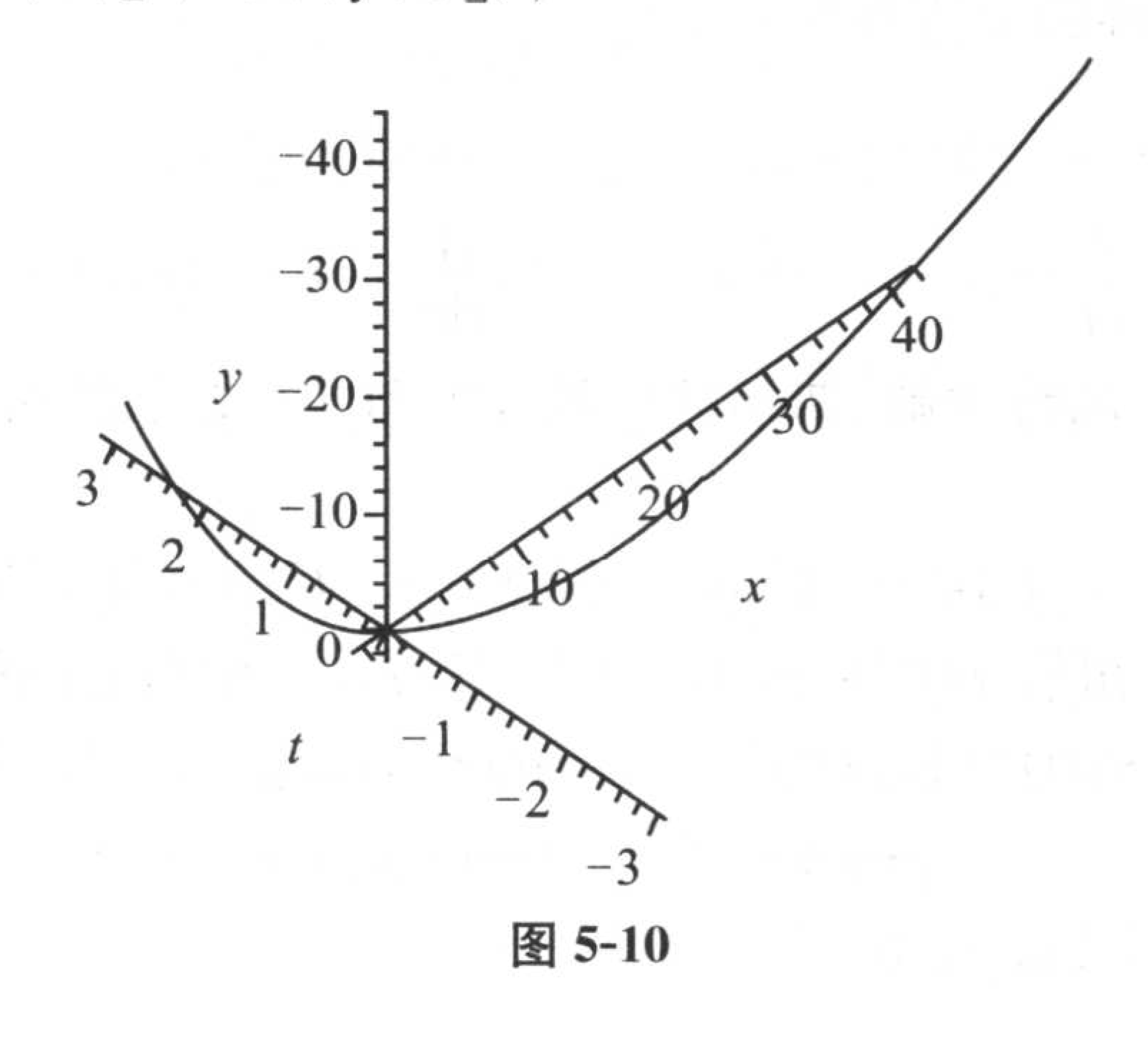


图 5-10

5.2　偏微分方程

5.2.1　偏微分方程的求解和作图命令

1. 偏微分方程求解命令 pdsolve

pdsolve 是求解偏微分方程(Partial Differential Equation)的常用命令，其用法有：

```
pdsolve(偏微方程,待解变量,选项)
pdsolve(偏微方程,初值或边界条件,选项)
```

pdsolve 为标准库函数，可直接使用。

2. 数值解作图命令 PDEplot

```
PDEplot(偏微方程,初值,参数范围,选项)
```

PDEplot 位于 PDEtools 函数包中，使用前必须先调出 PDEplot 函数包。

5.2.2　一阶拟线性和非线性微分方程

【例 11】　画出偏微分方程 $z_x + z \cdot z_y = 0$ 的解曲面 $z(x,y)$。

```
> pde：=D[1](z)(x,y)+z(x,y)*D[2](z)(x,y)=0;
```

$$pde := (D_1(z))(x,y) + z(x,y)(D_2(z))(x,y) = 0$$

给定初始条件 $z(0,y)=\text{sech}(y)$，$-5<y<5$，再画三维空间中的参数曲面(图 5-11)。

```
> PDEtools[PDEplot](pde,[0,y,sech(y)],y=-5..5);
```

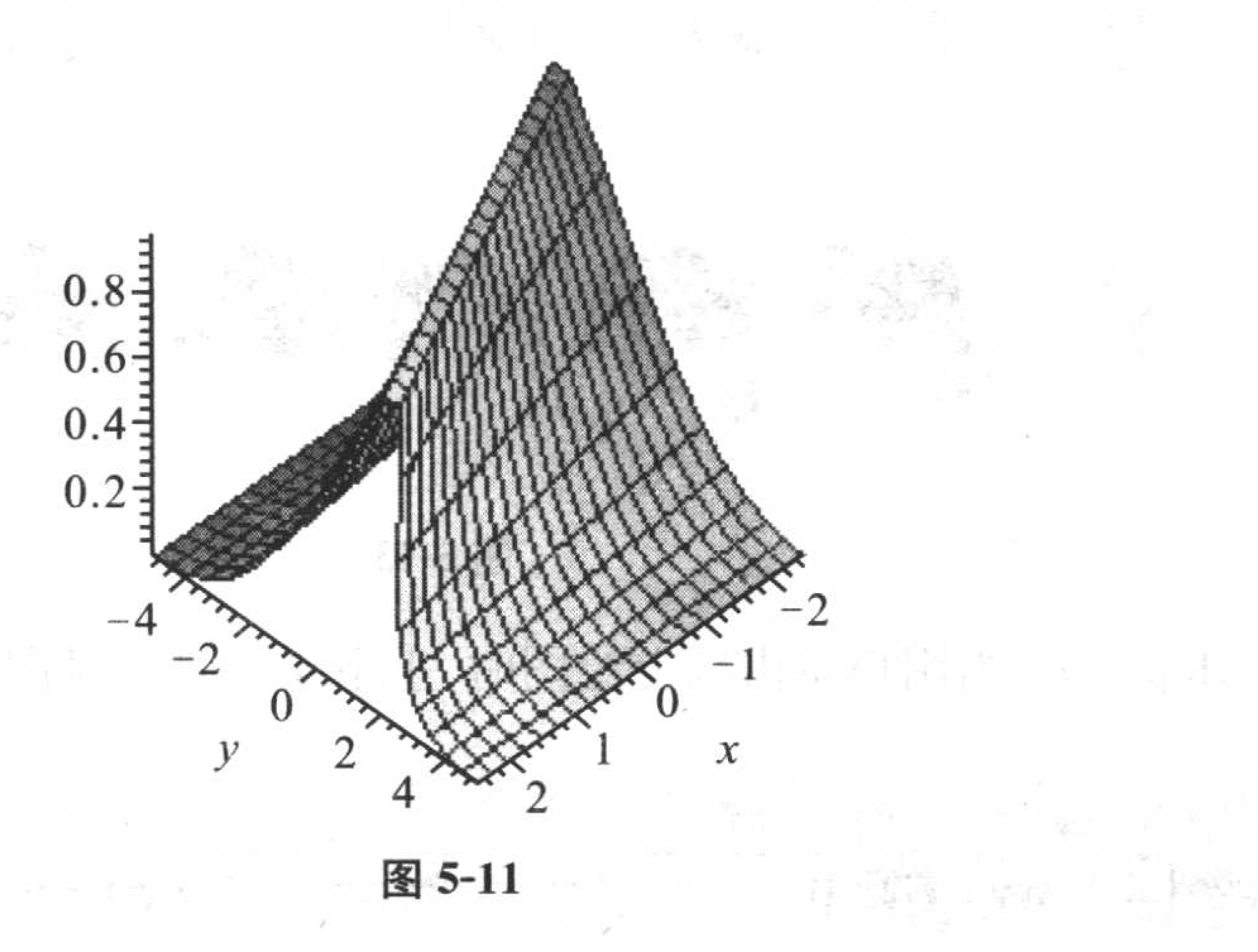


图 5-11

【例 12】　画出偏微分方程 $\cos(u_z)=u$ 的解曲面 $u(x,y,z)$。

```
> pde：=cos(diff(u(x,y,z),z))=u(x,y,z);
```

$$pde := \cos\left(\frac{\partial}{\partial z}u(x,y,z)\right) = u(x,y,z)$$

```
> PDEtools[PDEplot](pde,[cos(t)*sin(s),cos(t)*cos(s),cos(t),
  sin(t)],[t=0..Pi,s=0..Pi],numchar=[20,20]);
```

$u(x,y,z)$是四维空间中的三维超曲面。图 5-12 所示只是其一个 $z=z_0$ 截面 $u(x,y,z_0)$。

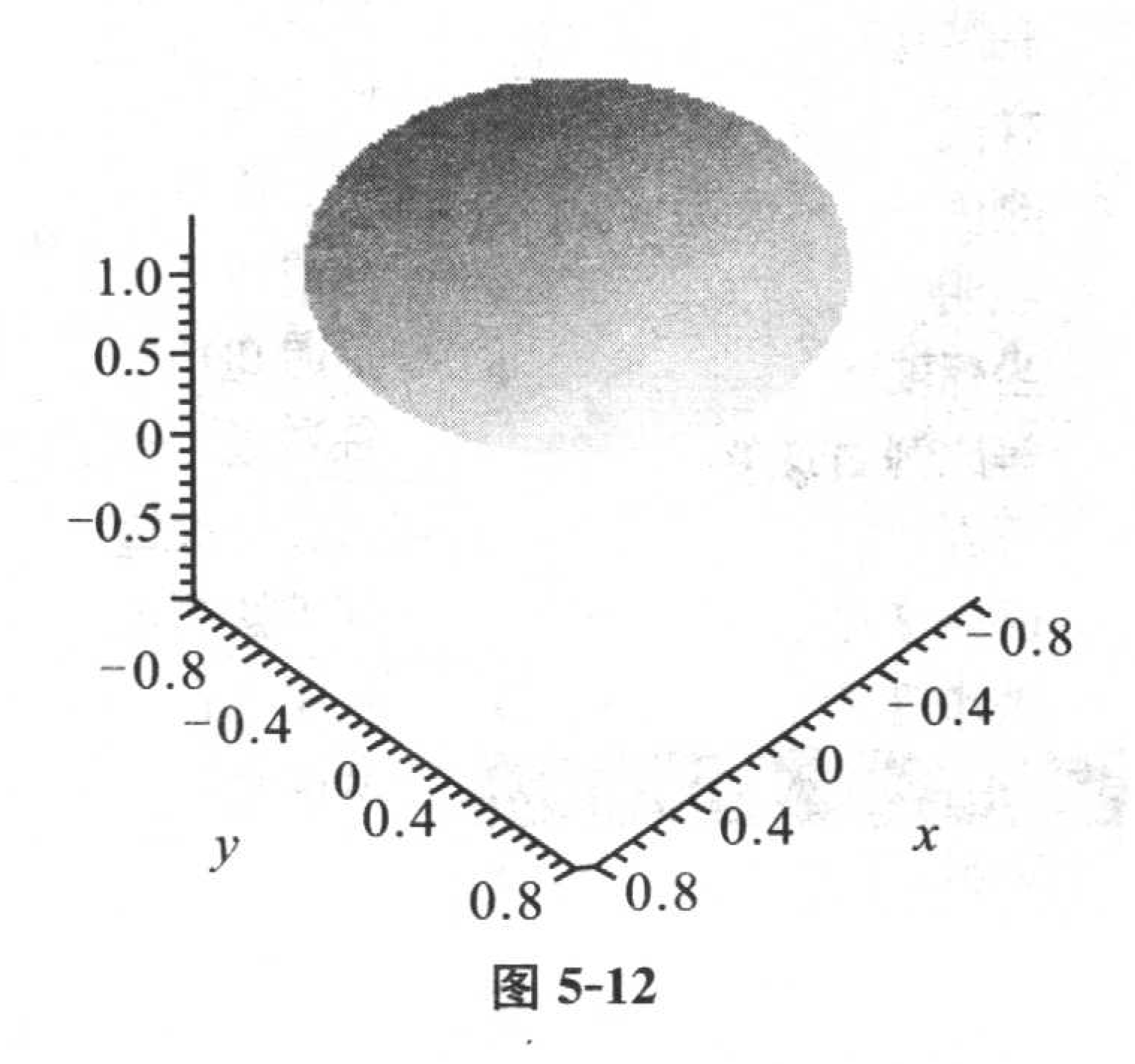


图 5-12

我们可以动画的方式观看 $u(x,y,z)$ 的其他 z 截面(图 5-13)。

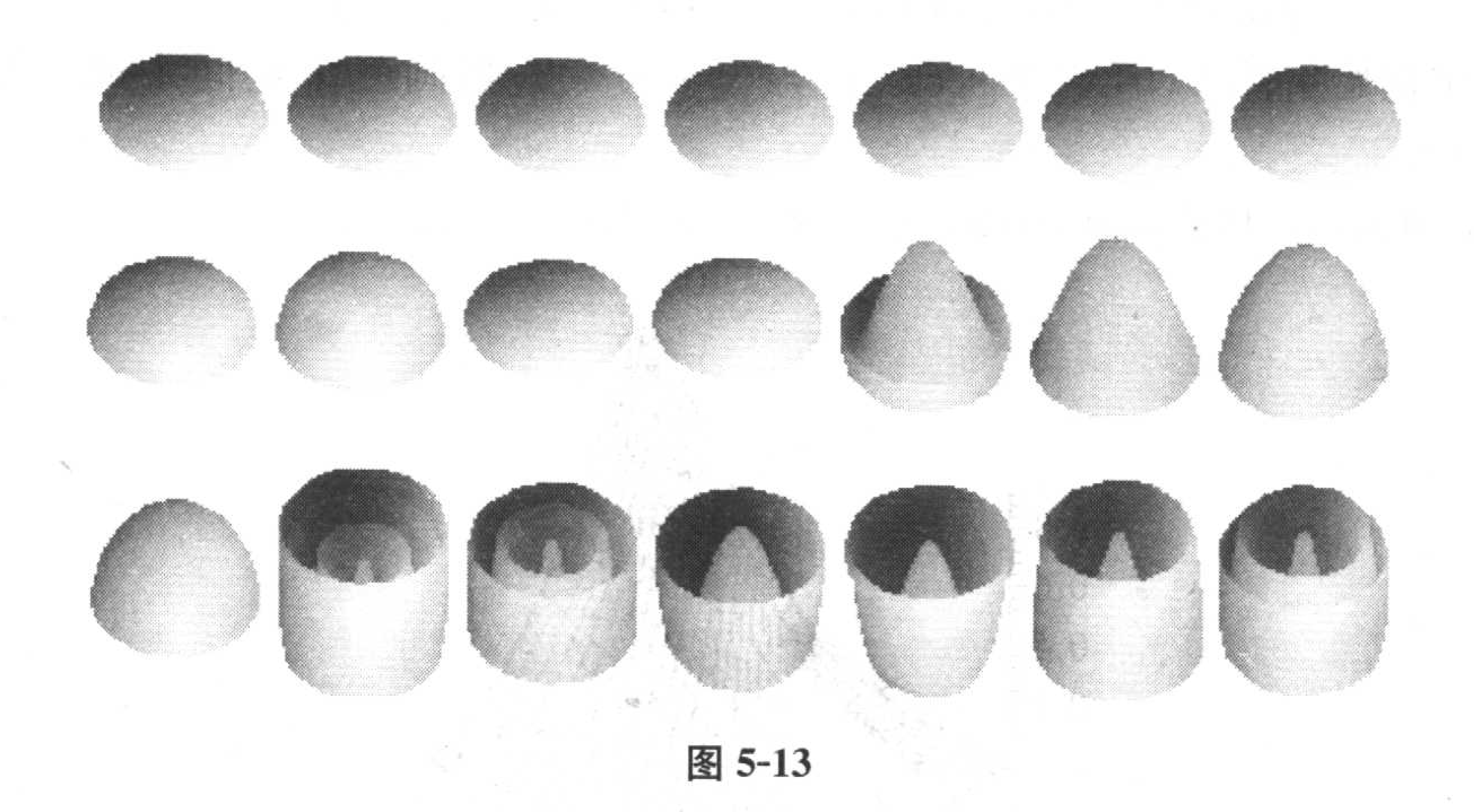


图 5-13

在 Maple 中,当用户单击以上绘图时,上下文工具栏见图 5-14。

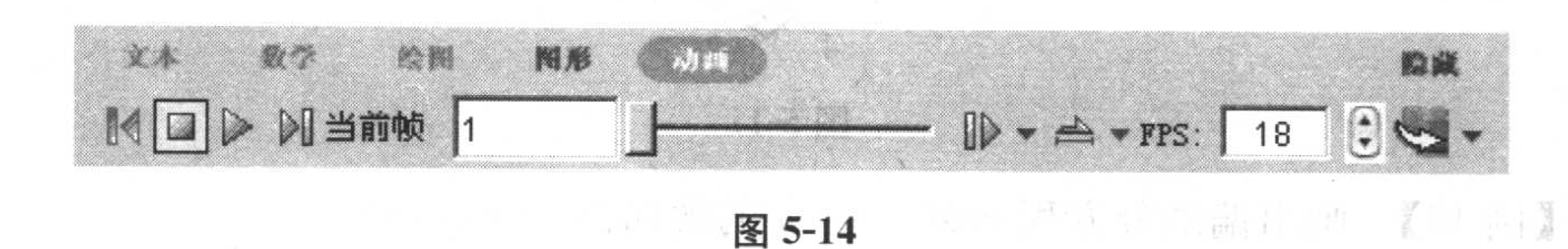


图 5-14

点击按钮 ▷ 就可以播放动画。另外我们也可以用鼠标右键单击绘图,弹出上下文菜单(图 5-15),选择菜单项 Animate-> Play 来播放动画。

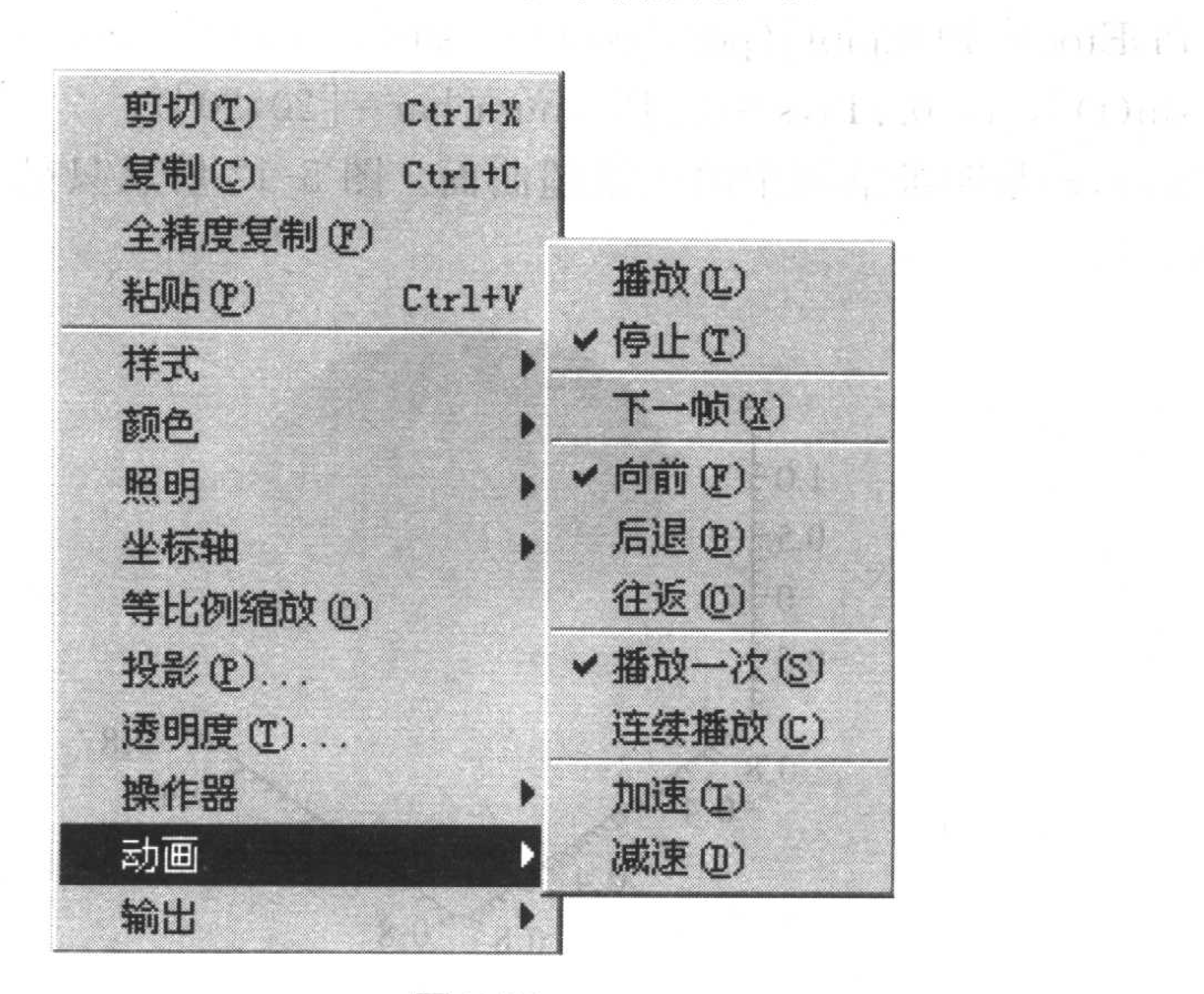


图 5-15

【例 13】 画出偏微分方程$(x^2+y^2+z^2)z_x-2xyz_y-2xz=0$的解曲面$z(x,y)$(图 5-16)。

```
> pde：=(x^2+y^2+z(x,y)^2)*diff(z(x,y),x)
  -2*x*y*diff(z(x,y),y)-2*x*z(x,y)=0;
```

$$pde := (x^2+y^2+z(x,y)^2)\left(\frac{\partial}{\partial x}z(x,y)\right)-2xy\left(\frac{\partial}{\partial y}z(x,y)\right)-2xz(x,y)=0$$

```
> PDEtools[PDEplot](pde,[t,t,sin(Pi*t/0.1)/10],t=0..0.1);
```

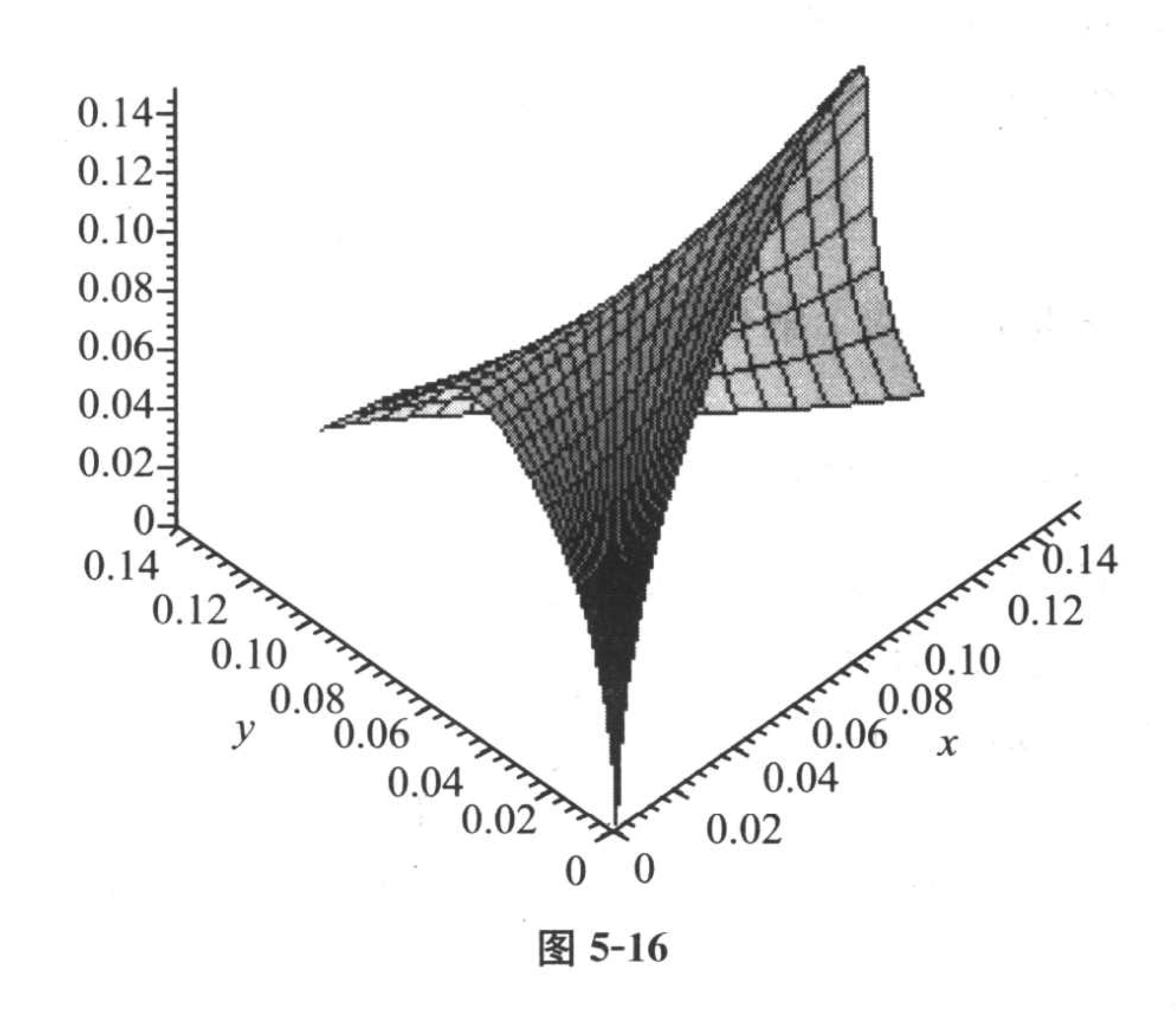


图 5-16

5.2.3 二阶微分方程

1. 波动方程

【例 14】 求解波动方程$u_{xx}=u_{tt}$。

```
> pdsolve(diff(u(x,t),x$2)=diff(u(x,t),t$2));
```

$$u(x,t)=_F1(t+x)+_F2(t-x)$$

这里给出了通解，其中_F1 和_F2 是任意两个具有二阶连续导数的一元函数。

2. 热传导方程

【例 15】 求解热传导方程$u_t-u_{xx}=0,t>0$。

```
> pdsolve(diff(u(x,t),t)=diff(u(x,t),x$2));
```

$$(u(x,t)=_F1(x)_F2(t)\ \&where\left[\left\{\frac{d^2}{dx^2}_F1(x)=_C1_F1(x),\frac{d}{dt}_F2(t)=_C1_F2(t)\right\}\right]$$

这里仅给出了分离变量形式的特解，其中_$F1$ 和_$F2$ 为满足给定方程的任意解，_C1 为常数。

3. 位势方程

位势方程的一般形式为：$-\Delta u = f(x)$，这是椭圆方程的典型代表。当 $f(x) \neq 0$ 时，称为 Poisson 方程；当 $f(x) = 0$ 时，称为调和方程，它的解称为调和函数。

【例 16】 求解 Poisson 方程 $u_{xx} + u_{yy} = -x$。

```
> pdsolve(diff(u(x,y),x$2)+diff(u(x,y),y$2)+x);
```

$$u(x,y) = _F1(y+\mathrm{I}x) + _F2(y-\mathrm{I}x) - \frac{1}{6}x^3$$

其中_$F1$ 和_$F2$ 是任意两个解析函数，$\mathrm{I}=\sqrt{-1}$。

【例 17】 求解 Cauchy-Riemann 方程 $\begin{cases} u_x = v_y \\ u_y = -v_x \end{cases}$。

```
> pdsolve({diff(u(x,y),x)-diff(v(x,y),y),
  diff(u(x,y),y)+diff(v(x,y),x)});
```

$$\{u(x,y) = _F1(y+\mathrm{I}x) + _F2(y-\mathrm{I}x),$$
$$v(x,y) = \mathrm{I}_F1(y+\mathrm{I}x) - \mathrm{I}_F2(y-\mathrm{I}x) + _C1\}$$

这里给出了复形式的通解，其中_$F1$ 和_$F2$ 是任意两个解析函数，_$C1$ 为常数，$\mathrm{I}=\sqrt{-1}$。

【例 18】 求解调和方程 $u_{xx} + u_{yy} = 0$。

```
> pdsolve(diff(u(x,y),x$2)+diff(u(x,y),y$2));
```

$$u(x,y) = _F1(y-\mathrm{I}x) + _F2(y+\mathrm{I}x)$$

同样的，_$F1$ 和_$F2$ 是任意两个解析函数，$\mathrm{I}=\sqrt{-1}$。

【例 19】 求解方程 $u_{xxyyy} = 0$。

这是一个高阶 PDE 问题。

```
> pdsolve(D[1,1,2,2,2](u)(x,y)); # D[1](u)表示对 u 的第一个变量求偏
  微分，D[1,1,2,2,2](U)表示对第一个变量求 2 次微分，对第二个变量求 3 次
  微分。
```

$$u(x,y) = _F5(x) + _F4(x)y + \frac{1}{2}_F3(x)y^2 + _F2(y) + _F1(y)x$$

这是方程的通解，其中_$F1$ 和_$F2$ 是任意两个具有 3 阶连续导数的一元函数，_$F3$、_$F4$ 和_$F5$ 是任意两个具有 2 阶连续导数的一元函数。

【例 20】 求解偏微分方程 $u_{xxyyy}=\sin(xy)$。

> pdsolve(D[1,1,2,2,2](u)(x,y)=sin(x*y));

$$u(x,y) = _F5(x) + _F4(x)y + \frac{1}{2}_F3(x)y^2 + _F2(y) + _F1(y)x$$

$$-x\left(-\frac{xy\mathrm{Si}(xy)+\cos(xy)}{x^2} + \frac{\frac{\frac{1}{2}x^2y^2Ci(xy) - \frac{1}{2}\cos(xy) - \frac{1}{2}xy\sin(xy)}{x} + \frac{\cos(xy)}{x}}{x}\right)$$

> expand(%);

$$u(x,y) = _F5(x) + _F4(x)y + \frac{1}{2}_F3(x)y^2 + _F2(y) + _F1(y)x$$

$$+ y\mathrm{Si}(xy) + \frac{1}{2}\frac{\cos(xy)}{x} - \frac{1}{2}xy^2Ci(xy) + \frac{1}{2}y\sin(xy)$$

【例 21】 求解二阶变系数方程 $\frac{\mathrm{d}^2}{\mathrm{d}x^2}r(x) + 2k^2\mathrm{sech}(kx)^2r(x) = 0$ 。

> eq1 := diff(r(x), x, x)+2*k^2*sech(k*x)^2*r(x);

$$eq1 := \left(\frac{\mathrm{d}^2}{\mathrm{d}x^2}r(x)\right) + 2k^2\mathrm{sech}(kx)^2r(x)$$

> dsolve(eq1);

$$r(x) = _C1\sqrt{\frac{(\cosh(kx)-1)(\cosh(kx)+1)}{\cosh(kx)^2}} +$$

$$\frac{1}{\left(\sqrt{\frac{(\cosh(kx)-1)(\cosh(kx)+1)}{\cosh(kx)^2}}\right)^{(3/2)}\cosh(kx)^4}\left(_C2\left(\left(\frac{-\cosh(kx)^2+1}{\cosh(kx)^2}\right)^{(3/2)}\cosh(kx)^4 -\right.\right.$$

$$\left.\left.\arctan\left(\frac{1}{\sqrt{-\frac{\cosh(kx)^2-1}{\cosh(kx)^2}}}\right)(\cosh(kx)-1)^2(\cosh(kx)+1)^2\right)\right)$$

充分展现了 Maple 求解微分方程的能力和水平。

习　　题

1. 求解一阶常微分方程。

(1) $y'=\frac{x^2}{y(1+x^3)}$　　(2) $y'=-y^2\sin x$

(3) $y'=(\cos x\cos 2y)^2$　　(4) $y'=\frac{x-\mathrm{e}^{-x}}{y+\mathrm{e}^y}$

2. 求解高阶线性常微分方程。

(1) $u''+5\left(u-\frac{1}{6}u^3\right)=0$　　(2) $u'''-2u''-3u'=0$

(3) $u''+6u'+9u=\cos 2x$　　(4) $(2x+1)^2u''-4(2x+1)u'+8u=0$

3. 解常微分方程组。

(1) $\begin{cases}x'=2x+3y\\y'=3x+4y\end{cases}$　　(2) $\begin{cases}x'=y-5\cos t\\y'=2x+y\end{cases}$　　(3) $\begin{cases}x'=2x-y+z\\y'=2x+2y-z\\z'=x+2y-z\end{cases}$

4. 求解一阶偏微分方程。

(1) $(y+z)u_x+(z+x)u_y+(x+y)u_z=0$

(2) $(x^2+y^2)u_x+6xyu_y=0$

(3) $(xy^3-2x^4)u_x+(3y^4-x^3y)u_y=9u(x^3-y^3)$

(4) $x^2u_x-y^2u_y=u$

5. 求解高阶偏微分方程。

(1) $(1+u_y^2)u_{xx}-3u_xu_yu_{xy}+(1+u_x^2)u_{yy}=0$

(2) $u_y+5uu_x+u_{xxx}=0$　　(3) $u_{tt}=9(u_{xx}+u_{yy}+u_{zz})$

6. 求解偏微分方程组。

(1) $\begin{cases}u_t+(1+\sin x)u_x+2v_x+x=0\\v_t+u=0\end{cases}$　　(2) $\begin{cases}u_t=uu_x+vv_y+x\\v_t=vu_y+uv_x+y\end{cases}$

第 6 章　Maple 作图

6.1　二维函数作图命令(plot)

6.1.1　二维函数作图

用 plot 命令可以画出一元函数在指定区间上的二维函数图形。其用法有：

plot(函数,变量名)
plot(函数,范围,选项)

范围和选项均可省略,缺省时系统自动选取最佳设置。最简单的 plot 语句为：

plot(f(x),x=a..b)

画出 $f(x)$ 在区间 $[a,b]$ 上的图像,其中 f 可为表达式或过程。

【例 1】　画出函数 $f(x)=\frac{1}{x}\sin x$ 在区间[0,20]上的图像(图 6-1)。

```
> plot(sin(x)/x,x=0..20);
```

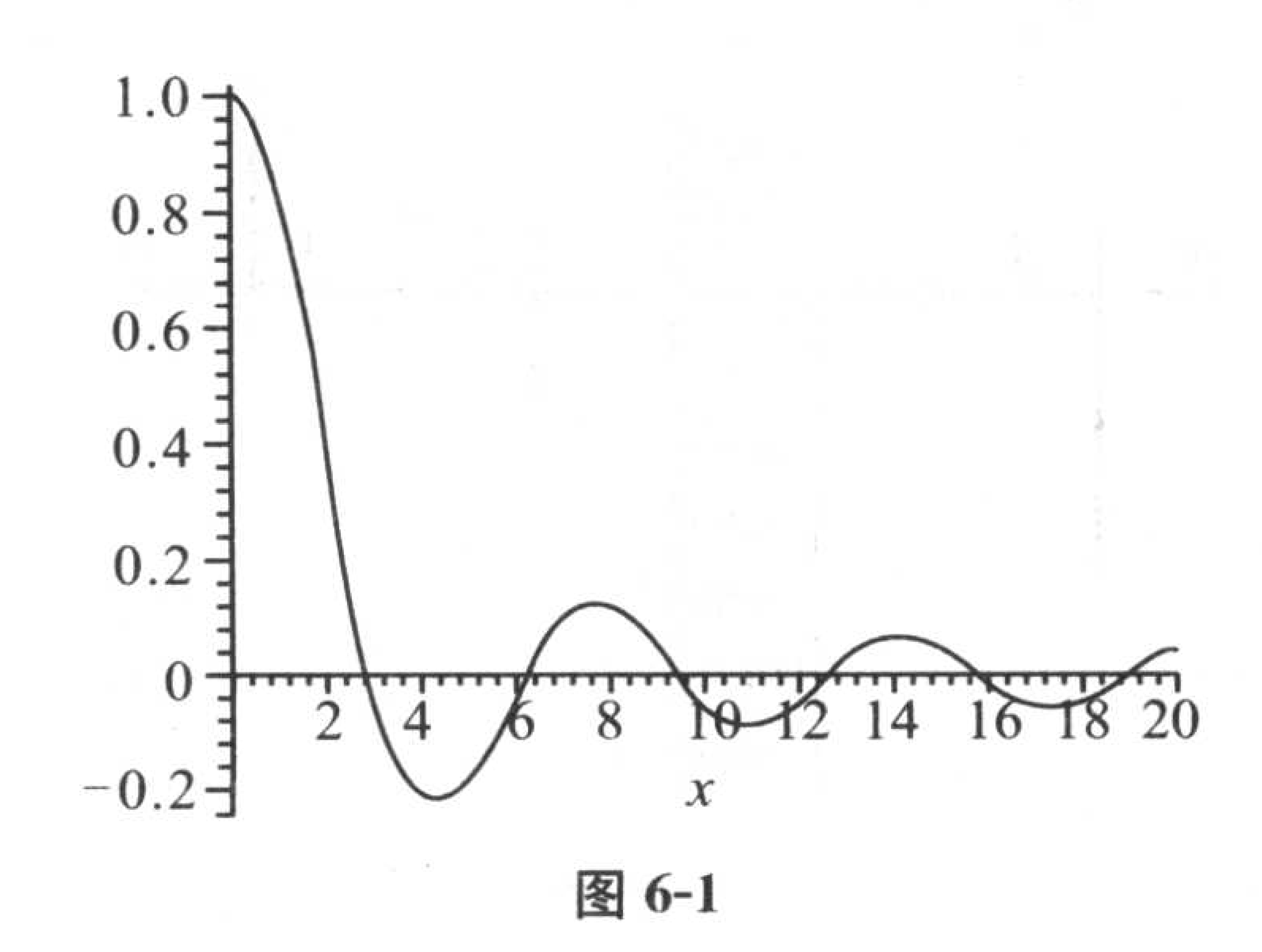


图 6-1

【例 2】 画出函数 $f(x) = 5x^2 - 8$ 在区间[−5,5]上的图像(图 6-2)。

```
> plot(5 * x^2-8,x=-5..5);
```

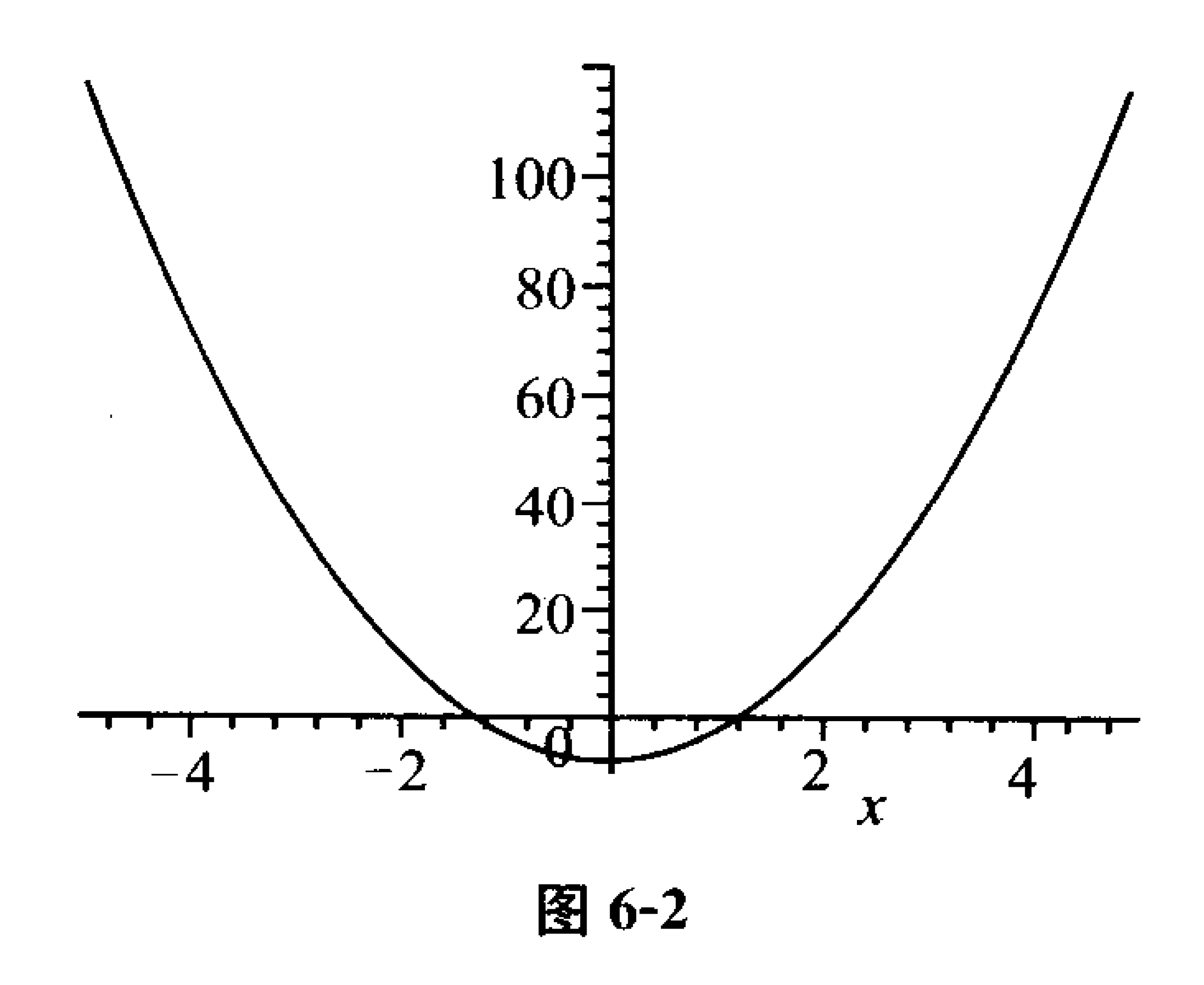


图 6-2

还可以限制 y 的取值范围(图 6-3)。

```
> plot(5 * x^2-8,x=-5..5,y=-20..40);
```

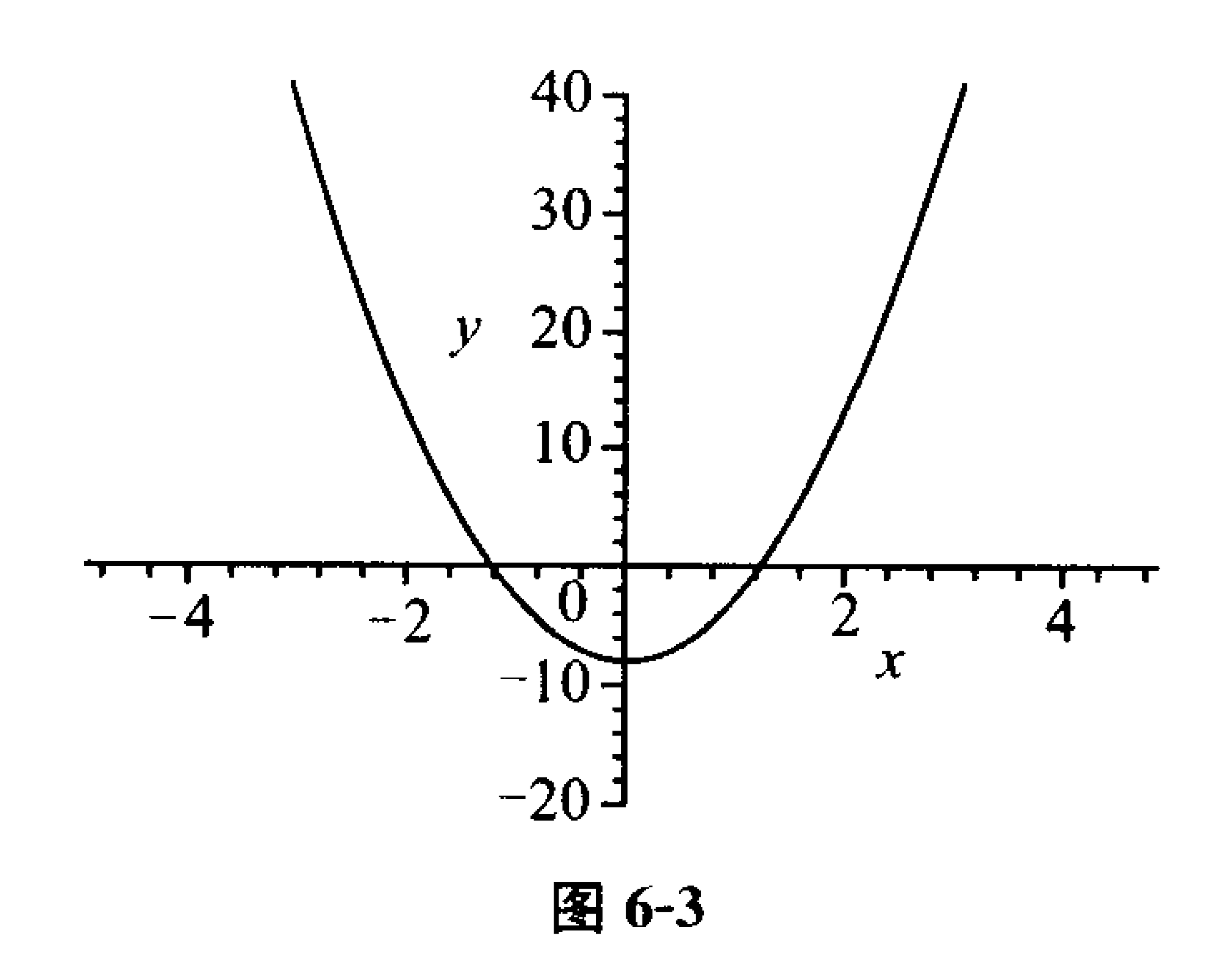


图 6-3

【例 3】 画出函数 $f(x) = \tan x$ 在区间[−6,6]上的图形(图 6-4)。

```
> plot(tan(x),x=-6..6);
```

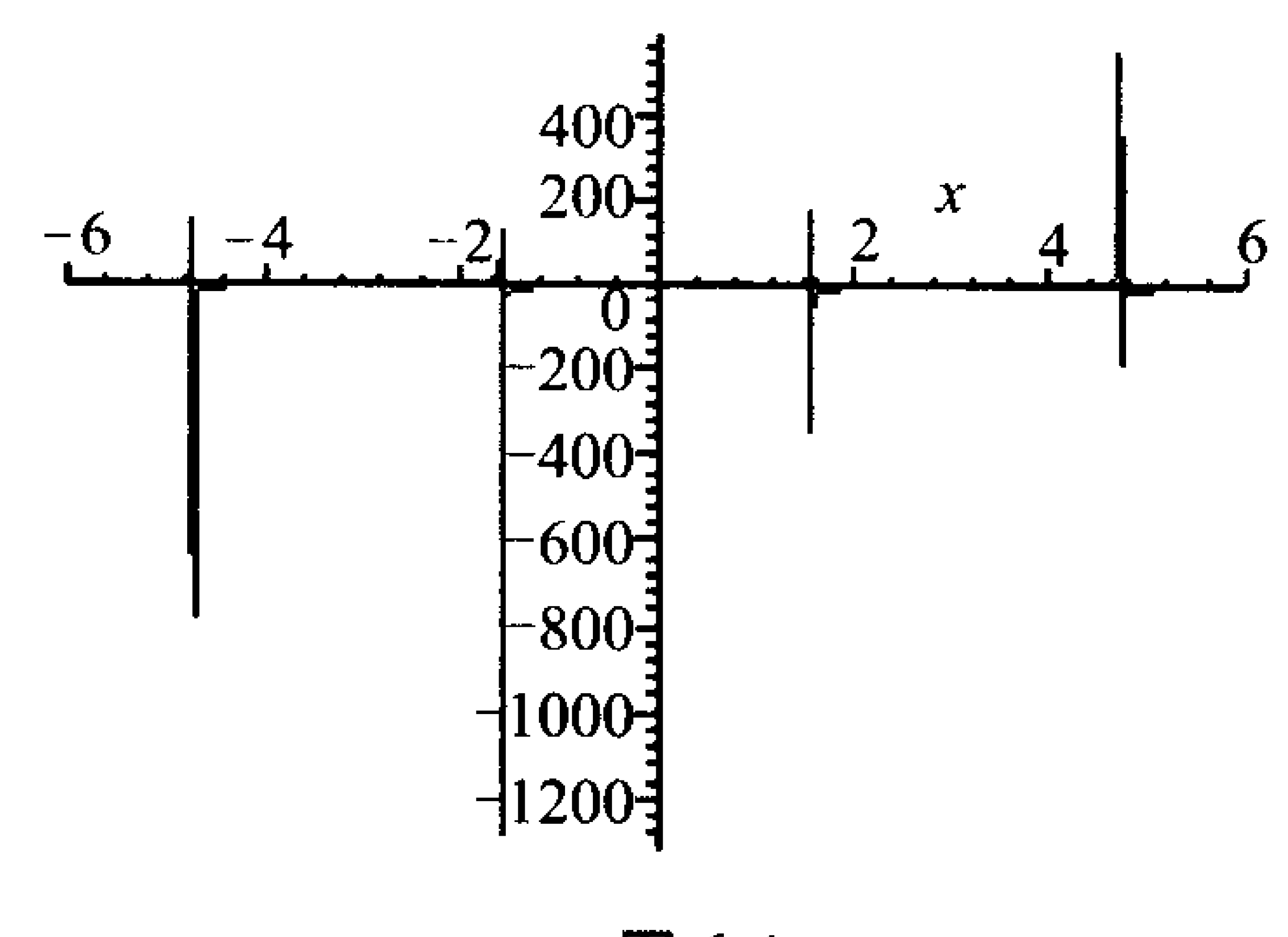


图 6-4

因为函数在部分点的值太大了，掩盖了那些有限值的函数点。这时，给函数值一个范围，便可以看到函数的轮廓(图 6-5)。

```
> plot(tan(x),x=-6..6,y=-6..6);
```

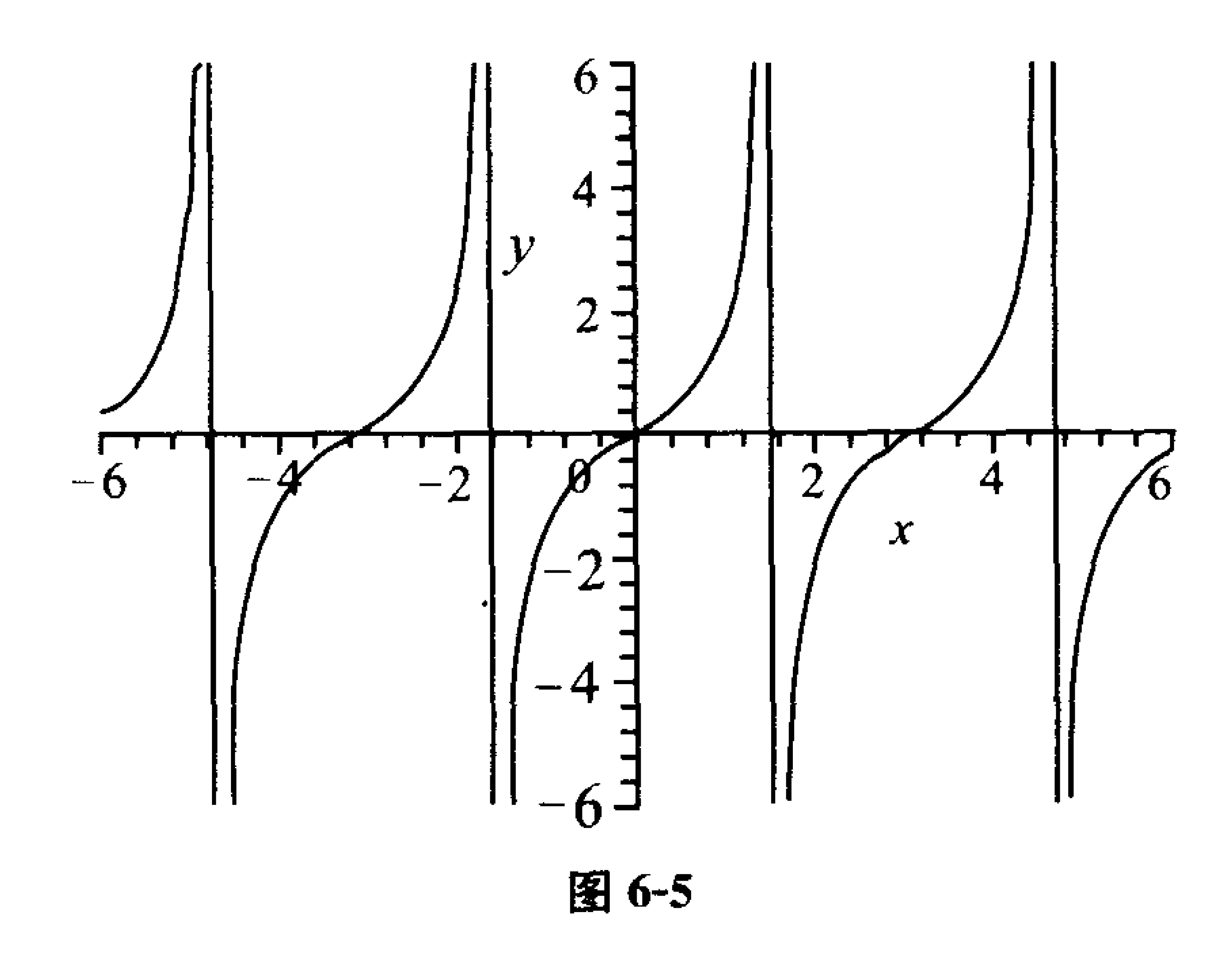


图 6-5

函数的取值区间可包含无穷远点，在无穷远点附近系统给出函数的示意图。

【例 4】 画出函数 $f(x)=\frac{1}{x}\sin x$ 在区间$(-\infty,\infty)$上的图形(图 6-6)。

```
> plot(sin(x)/x,x=-infinity..infinity);
```

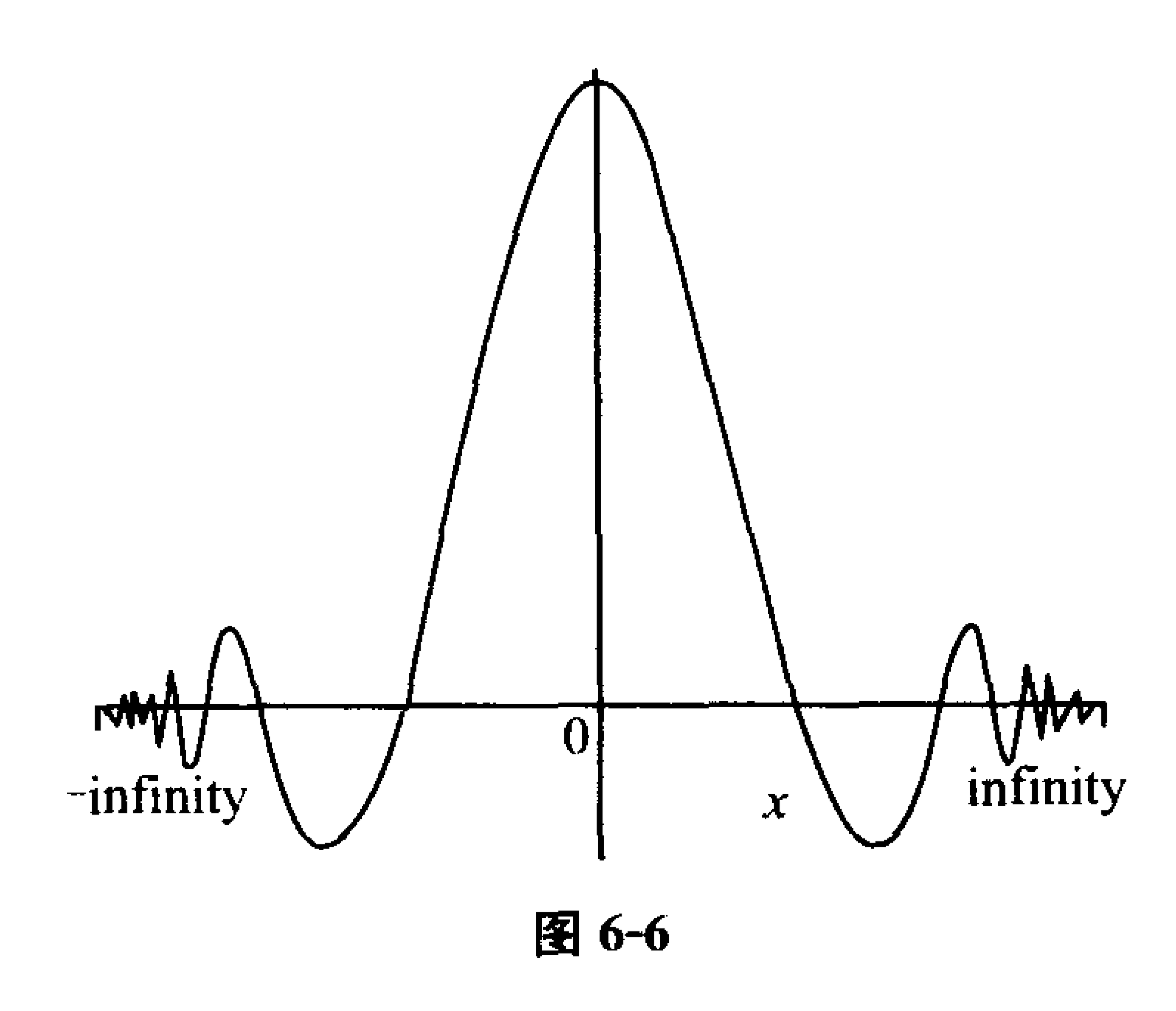


图 6-6

如果在绘图中没有给定函数的定义域，系统将取默认值[-10,10]为函数的定义域。

【例 5】 画出函数 $f(x) = x\cos x$ 的图像(图 6-7)。

```
> plot(x * cos(x),x); # 取绘图区间[-10,10]
```

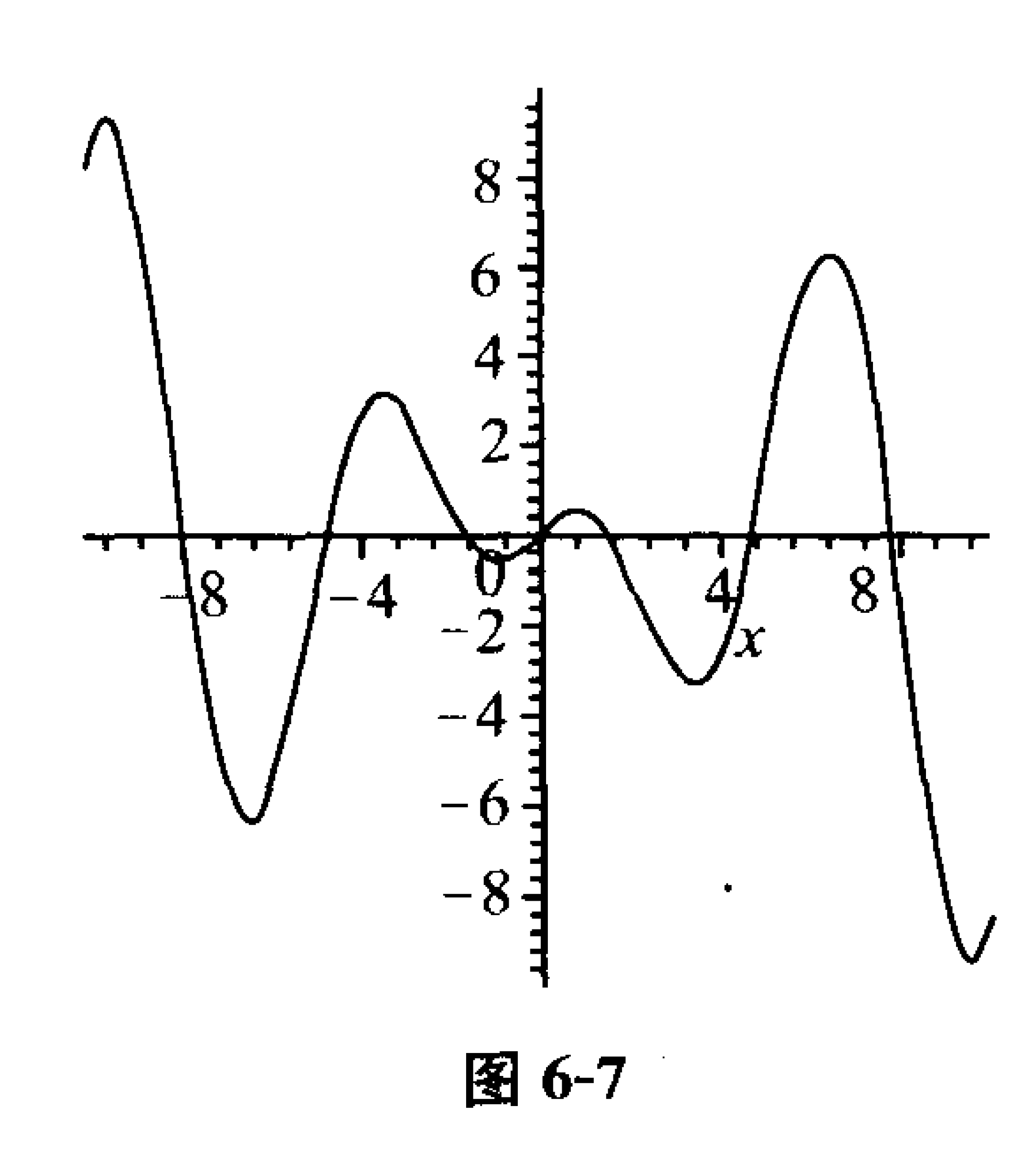


图 6-7

对于表达形式较为复杂的函数,可以定义过程,再画出函数图像。

【例 6】 画出函数 $f(x) = x^3\sin x + \frac{1}{3}x^2 + x\cos x$ 在区间[-3,3]上的图像(图 6-8)。

```
> f: =x-> x^3 * sin(x)+x^2/3+x * cos(x): plot(f(x),x=-3..3);
```

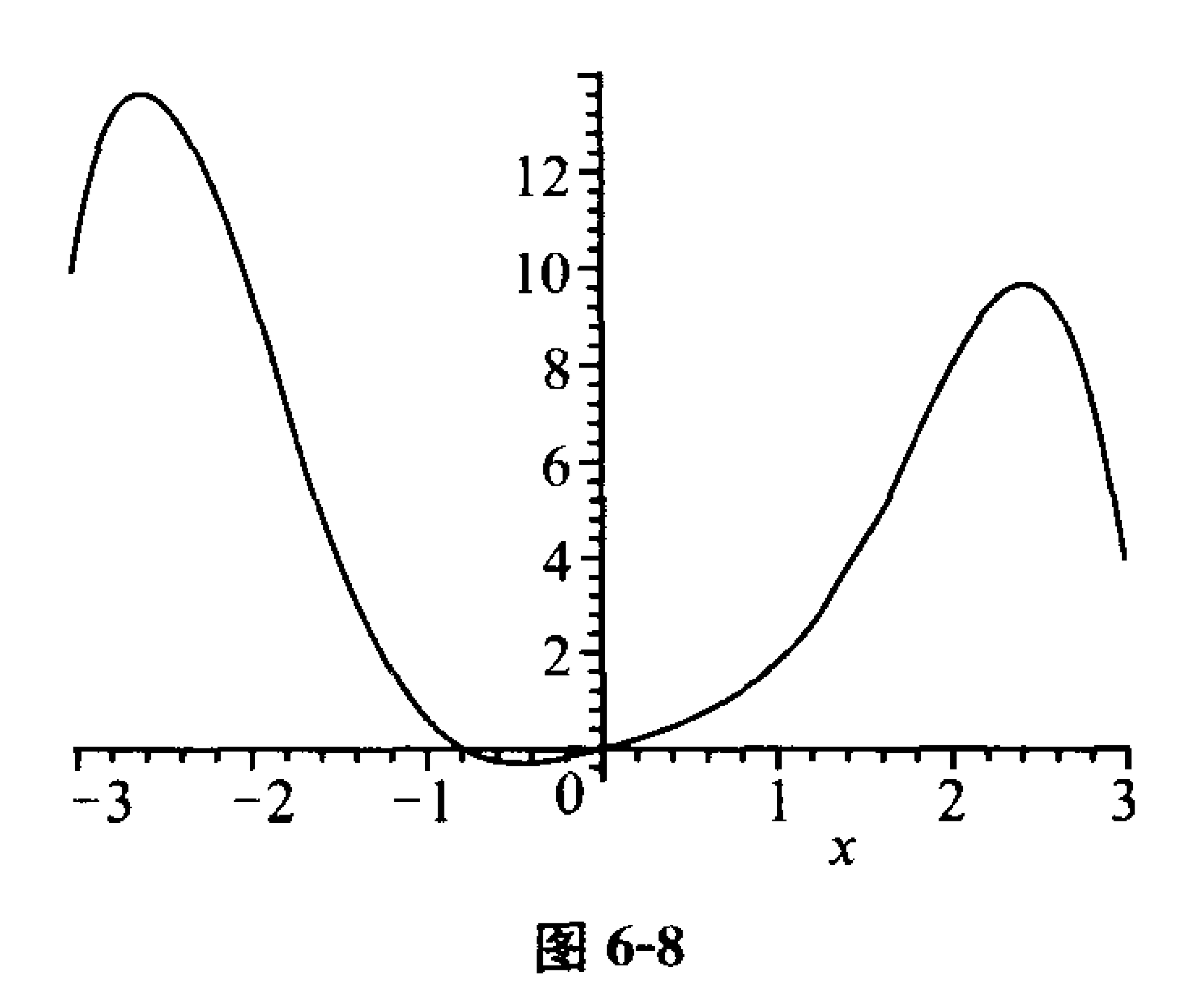


图 6-8

【例 7】 画出分段函数 $f(x) = \begin{cases} -x-\pi, & x < -\pi \\ \sin x, & -\pi \leqslant x \leqslant \pi \\ \frac{1}{2}(x-\pi), & x > \pi \end{cases}$,在区间[-6,6]上的图像(图 6-9)。

```
> f: =x-> piecewise(x<-Pi,-x-Pi,x<=Pi and x>=-Pi,
sin(x),x>Pi,(x-Pi)/2):
```

plot(f(x),x=-6..6);

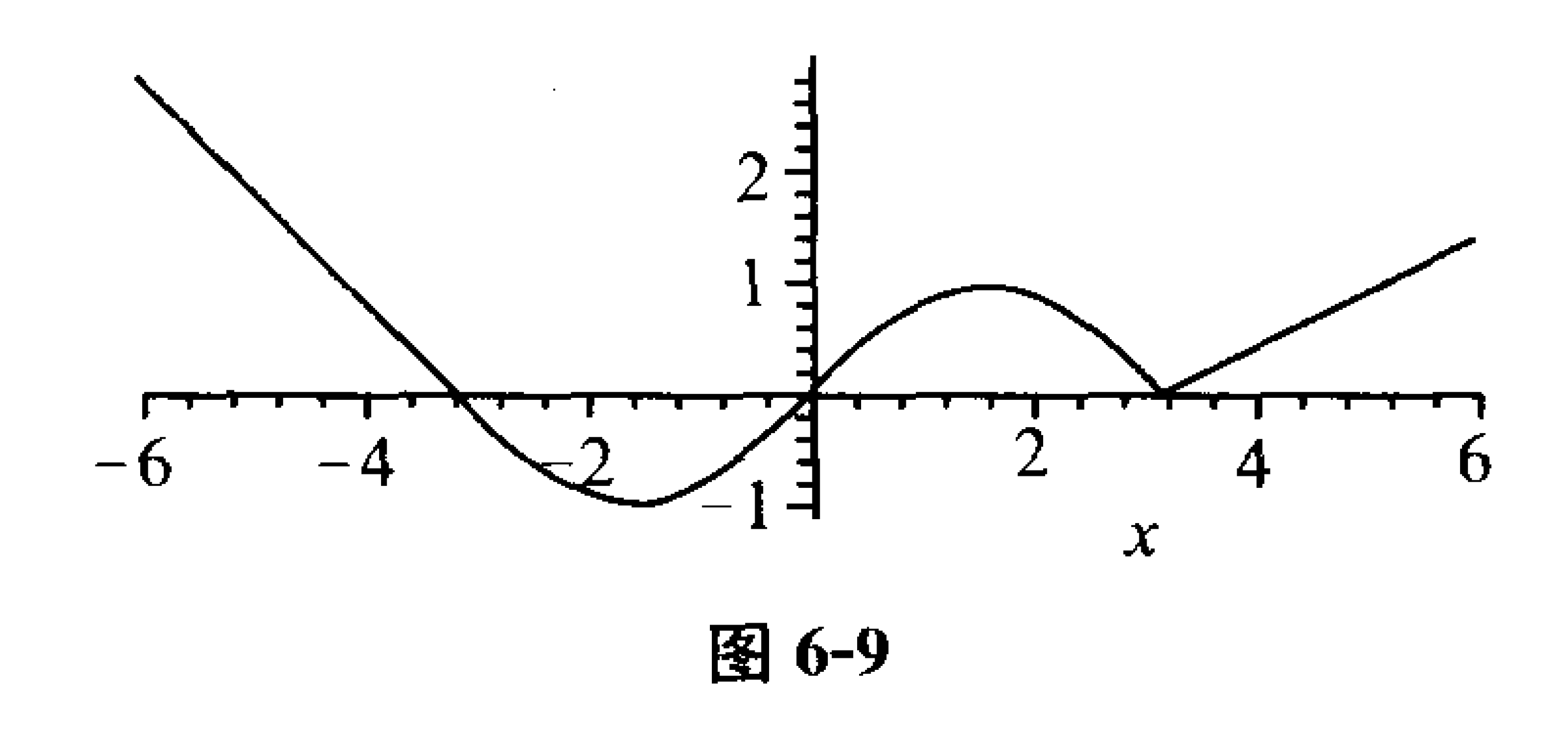


图 6-9

【例 8】 将一组给定点列用线段连接(图 6-10)。

\> plot([[0,0],[1,1],[2,0.5],[3,4],[5,5]]);

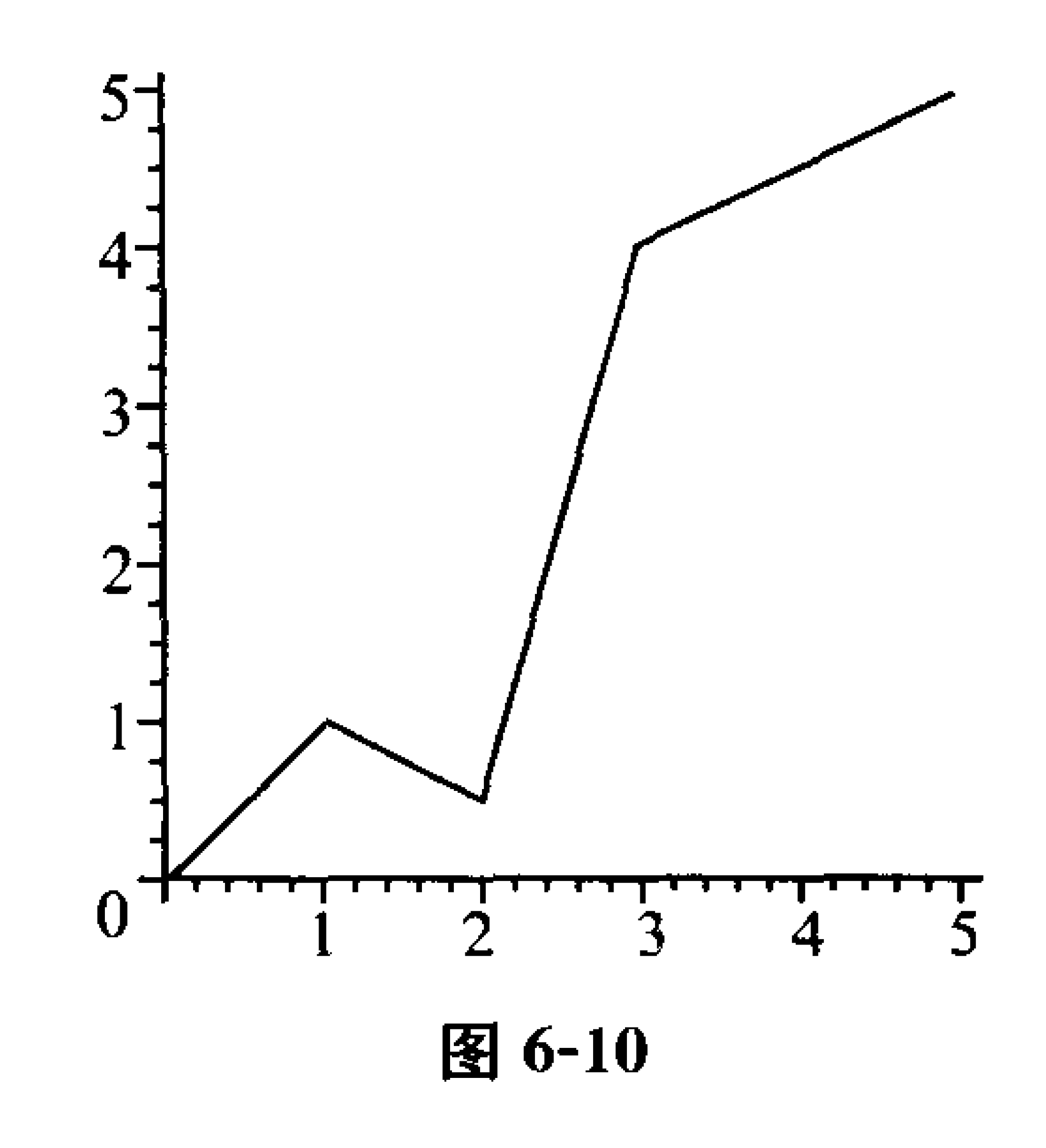


图 6-10

如果要在区间上同时画多个函数图形,可使用函数列表。

【例 9】 画出函数 $f(x)=-x^2$,$g(x)=x^2\sin(3x)$,$h(x)=x^2$ 在区间[-15,15]上的图像(图 6-11)。

\> plot([-x^2,x^2*sin(3*x),x^2],x=-15..15);

\# 系统自动对曲线分别染以红、绿、黄

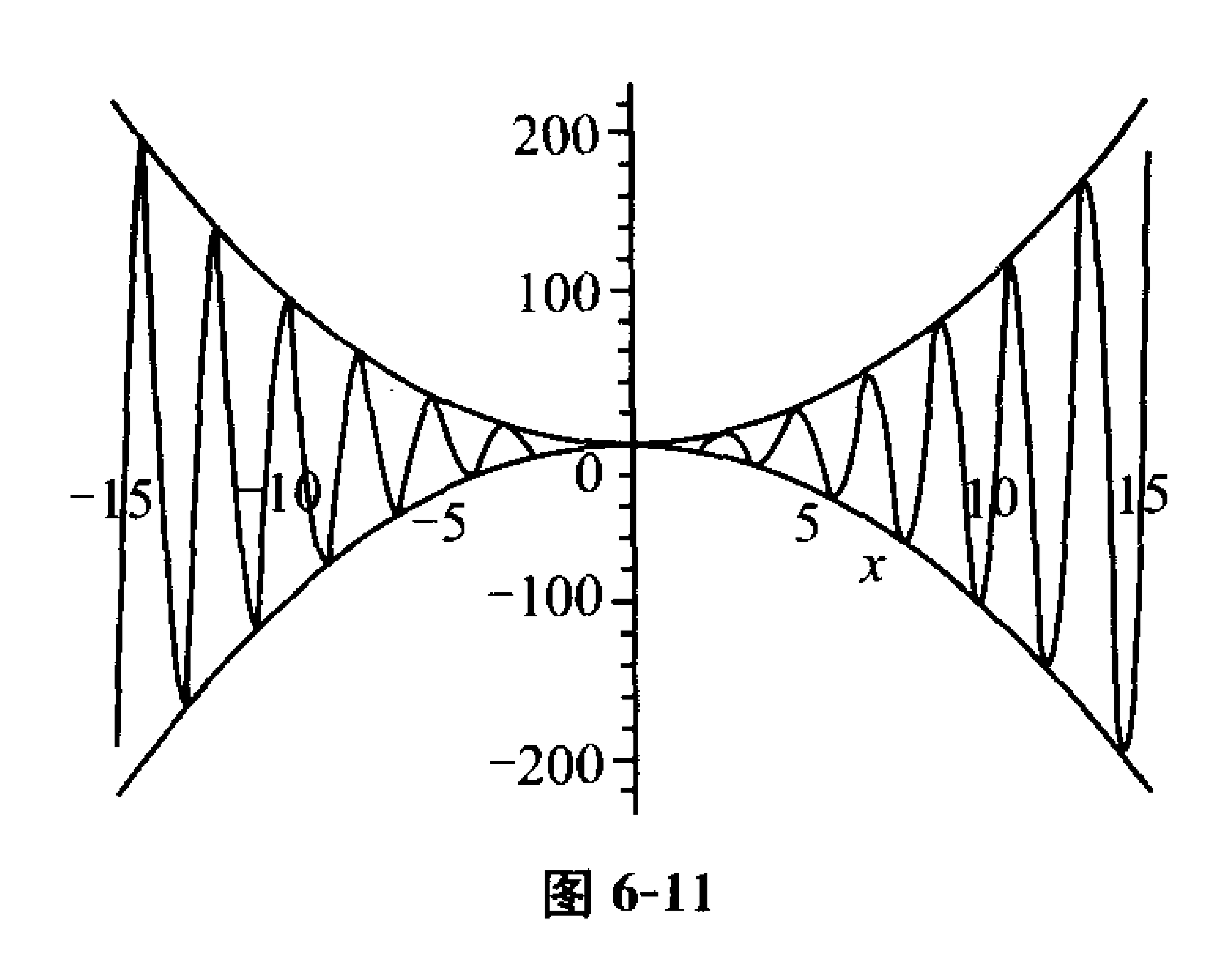


图 6-11

【例 10】 分别以样式“绿色点状”、“蓝色线状”画出函数 $f(x)=\sin x$,

$g(x) = \cos x$ 的图像(图 6-12)。

```
> plot([sin(x),cos(x)],x=0..2*Pi,color=[green,blue],
  style=[point,line]);
```

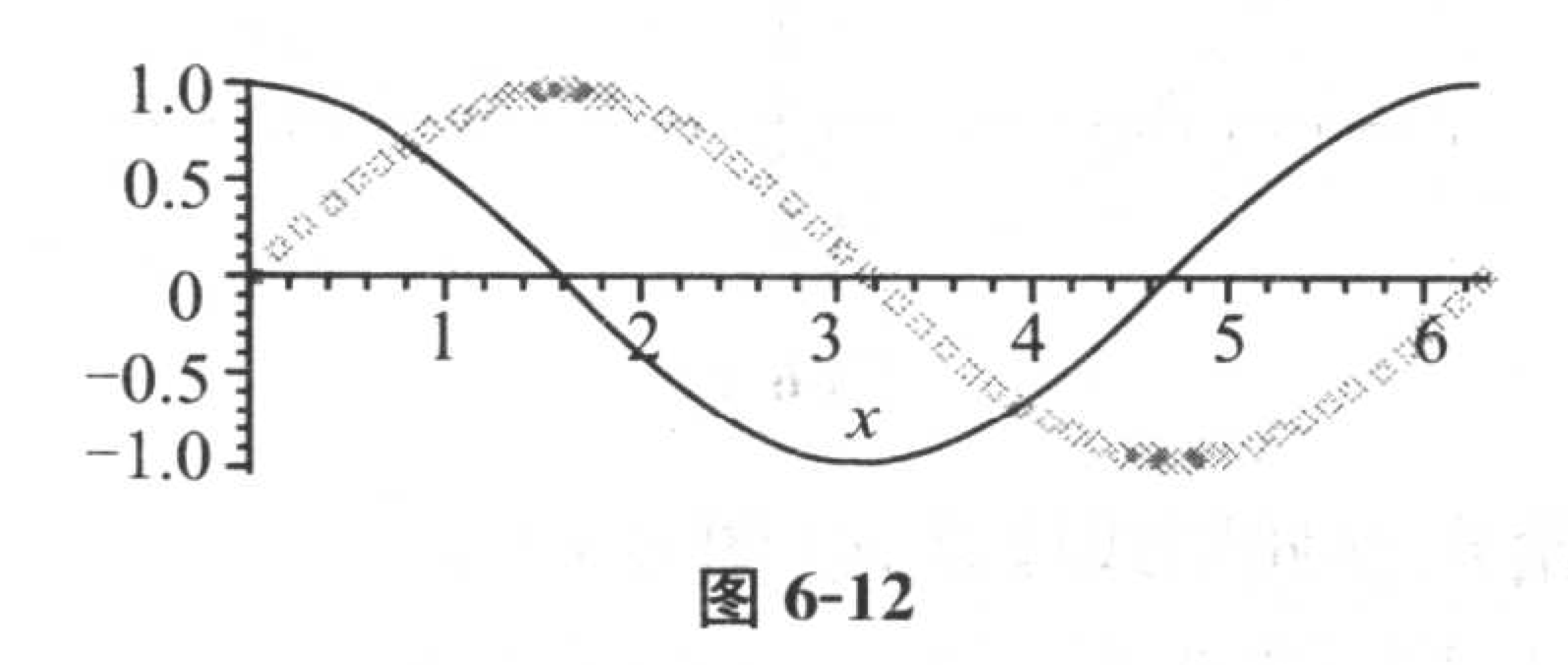


图 6-12

我们也可以先画出两个函数的图像,然后用鼠标右键点击某一函数图像,通过上下文工具栏(图 6-13)、上下文菜单(图 6-14)或图形菜单(图 6-15)来修改图像的属性。这等价于在 plot 函数中设置选项。

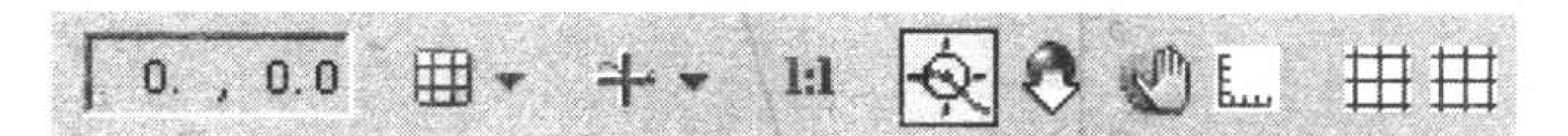


图 6-13 上下文工具栏

剪切(T) Ctrl+X
复制(C) Ctrl+C
全精度复制(P)
粘贴(P) Ctrl+V
样式
点
线样式
颜色
透明度
坐标轴
图例(L)
标题(T)
等比例缩放(N)
操作器
输出(E)

图 6-14 上下文菜单

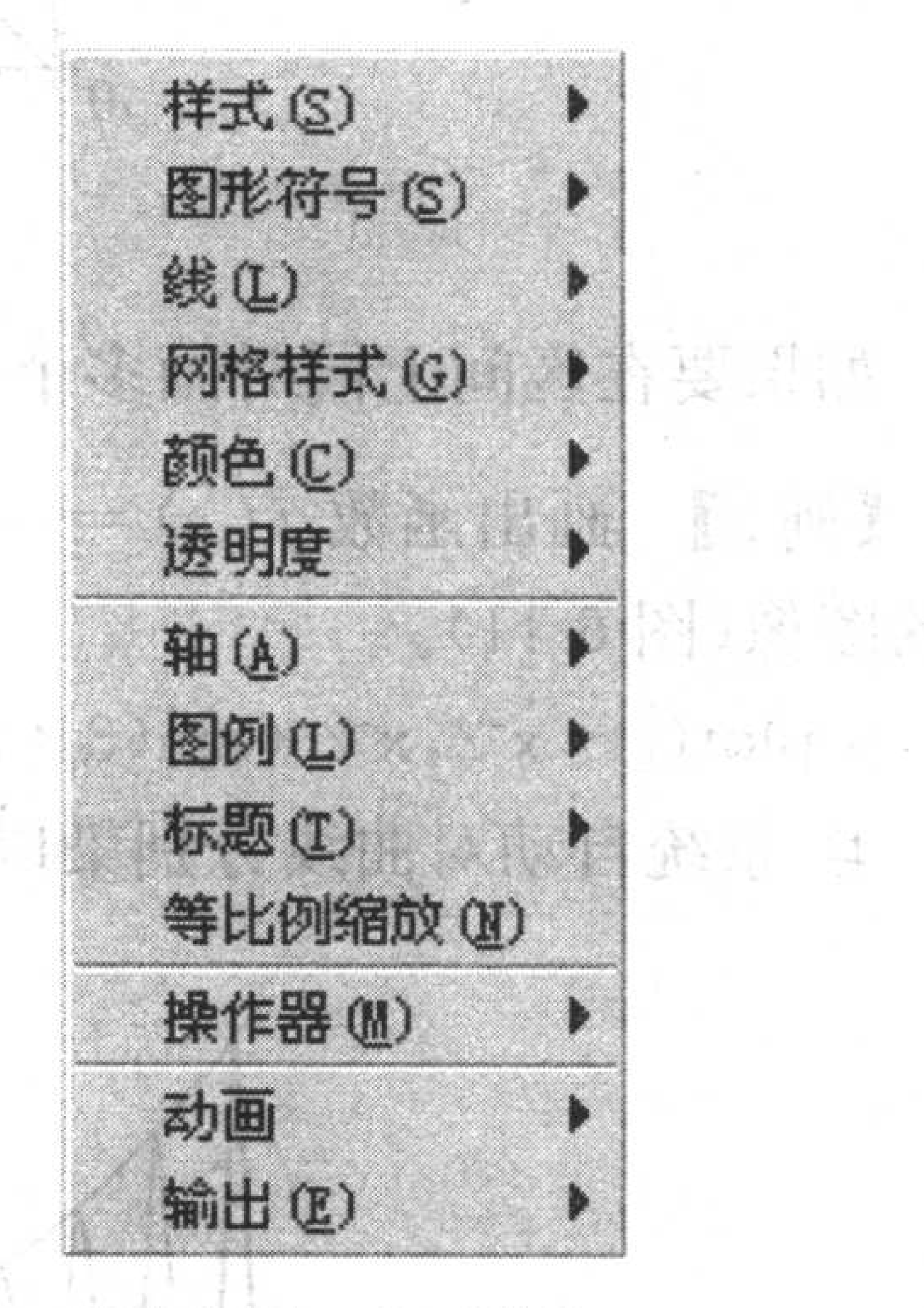


图 6-15 plot 菜单

6.1.2 plot 命令选项

下列为 plot 命令的部分选项及其意义。

style

设置曲线样式，取值 line、point、patch、pathnogrid。默认值是 line，将点用折线相连；point 按点状形式；patch 在包含面的情况下显示面；patchnogrid 与 patch 类似，但不显示网格线。

symbol

设置点的显示方式，取值 box、cross、circle、point、diamond。当 style=point 时，此设置有效。

symbolsize

设置点的大小。当 style=point 且 symbol 不为 point 时，此设置有效。

linestyle

设置线的显示方式，当 style=line 时，此设置有效。选项取值 SOLID、DOT、DASH、DASHDOT。也可以用数字 1、2、3、4 表示。

thickness

设置线的宽度，取值正整数。

axes

设置坐标轴的显示方式，取值 boxed、framed、normal、none。默认值是 normal。

scaling

设置图形的纵横比，取值 constrained、unconstrained。当 scaling=constrained 时，图形按实际大小比例显示；当 scaling=unconstrained 时，图形随画布的大小自动调整纵横比。

coords

设置函数的坐标系，取值 bipolar、cardioid、cassinian、elliptic、hyperbolic、invcassinian、invelliptic、logarithmic、logcosh、maxwell、parabolic、polar、rose、tangent。默认值为直角坐标系。

discont

当 discont=true 时，系统先判断函数是否连续，然后将绘图区域分为几个连续的区间后绘图，默认值是 false。

view

设置图形的视野范围，view=[xmin..xmax,ymin..ymax]。

title

设置图形的标题，title=字符串。

legend

设置图例。当同时显示多条曲线时，legend=字符串列表，分别对每条曲线加以说明。

labels

设置坐标轴的标注。labels=字符串列表。

labeldirections

设置坐标轴的标注的书写方向，取值 horizontal、vertical。

numpoints

设置图形的采样点个数。默认值是 50 个。

sample

设置图形的初始采样点。sample=采样点序列。当 adaptive=false 的时候，此选项明确地控制图形的表现。

adaptive

默认的采样点由 numpoints 和 sample 控制。adaptive=true 将在曲线弯曲的地方细分采样点，以提高图形的精度。adaptive=false 将取消局部的细分。

filled

设置是否用颜色填充曲线与 x 轴之间的部分，取值 true 或 false。

tickmarks

设置坐标轴刻度的个数，tickmarks=[m,n]，m 和 n 必须是正整数，也可以取 default。

xtickmarks

设置 x 轴刻度的个数。

ytickmarks

设置 y 轴刻度的个数。

color

设置曲线颜色。color=颜色名称或颜色值。表 6-1 列出了 24 种常用的颜色选项。

表 6-1 常用的颜色选项

选 项	颜 色	选 项	颜 色
aquamarine	碧绿色	black	黑色
blue	蓝色	brown	棕褐色
coral	珊瑚色	cyan	蓝绿色
gold	金黄色	gray 或 grey	灰色
green	绿色	khaki	黄褐色
magenta	红紫色	maroon	栗色
navy	海蓝色	orange	橙色

续表

选　项	颜　色	选　项	颜　色
pink	粉红色	plum	暗紫色
red	红色	sienna	土黄色
tan	棕黄色	turquoise	青绿色
violet	紫罗兰色	wheat	淡黄色
white	白色	yellow	黄色

【例 11】 请比较 constrianed 和 unconstrained 选项效果(图 6-16,6-17)。

```
> plot(sqrt(1-x^2),x=-1..1,scaling=constrained); # 按原比例绘图
```

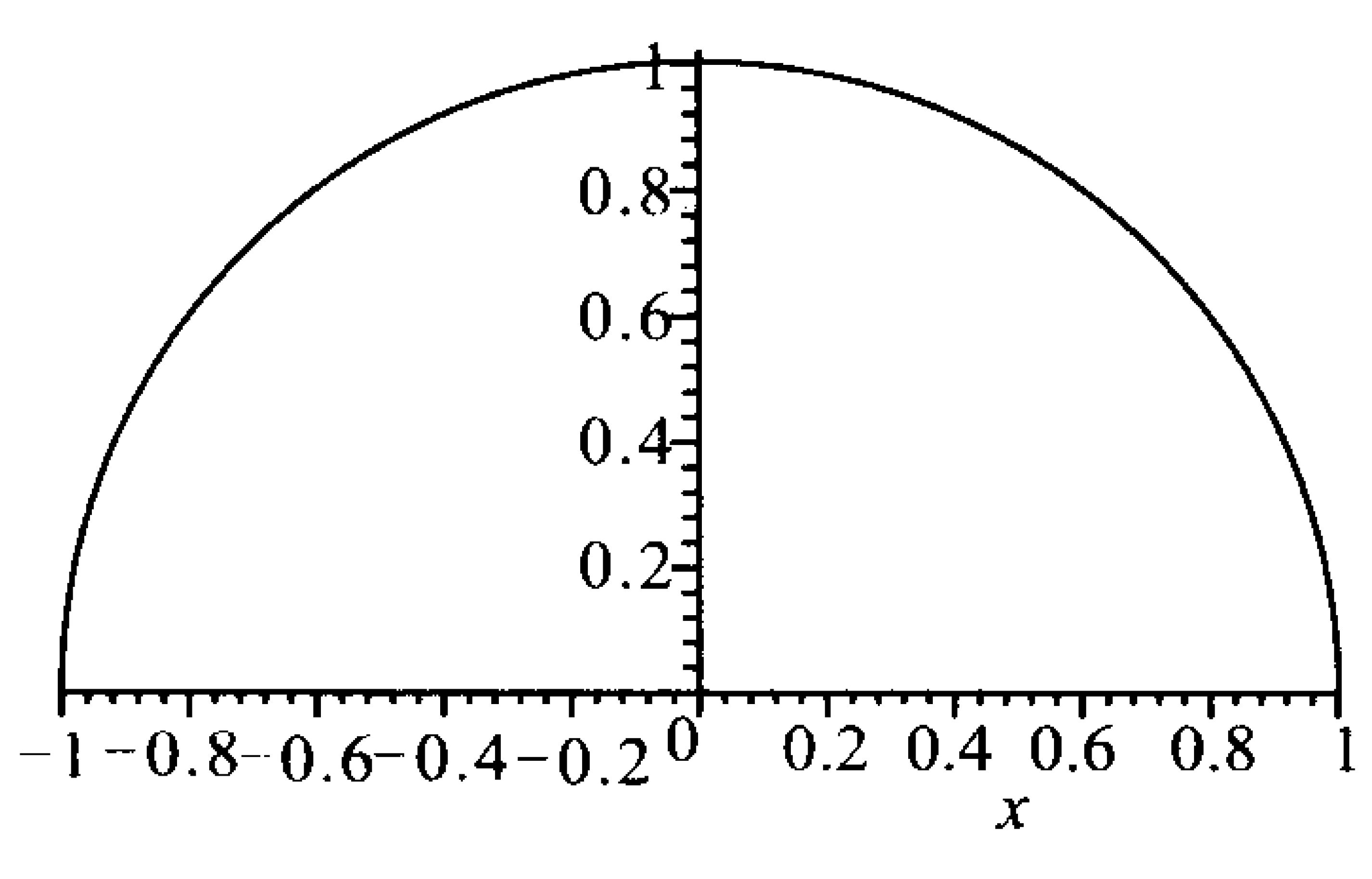


图 6-16

```
> plot(sqrt(1-x^2),x=-1..1); # 图形比例由画布大小决定
  或 plot(sqrt(1-x^2),x=-1..1,scaling=unconstrained);
```

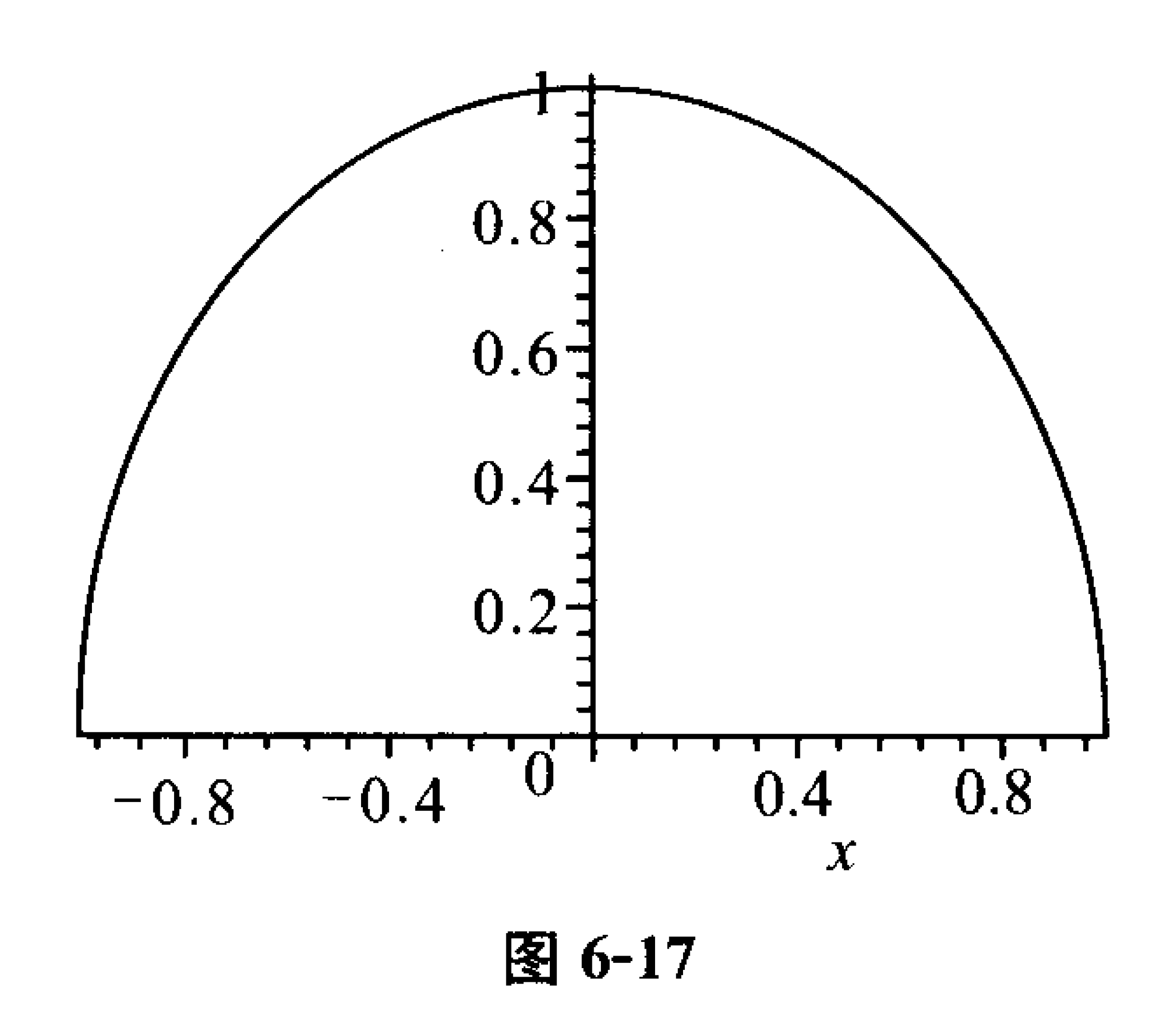


图 6-17

【例 12】 设置曲线的颜色、标题和图例(图 6-18)。

```
> plot([sin(x),cos(x)],x=-Pi..Pi, title="Simple Trig Functions",
  color=[red,blue],style=[line,point],legend=["Sine","Cosine"]);
```

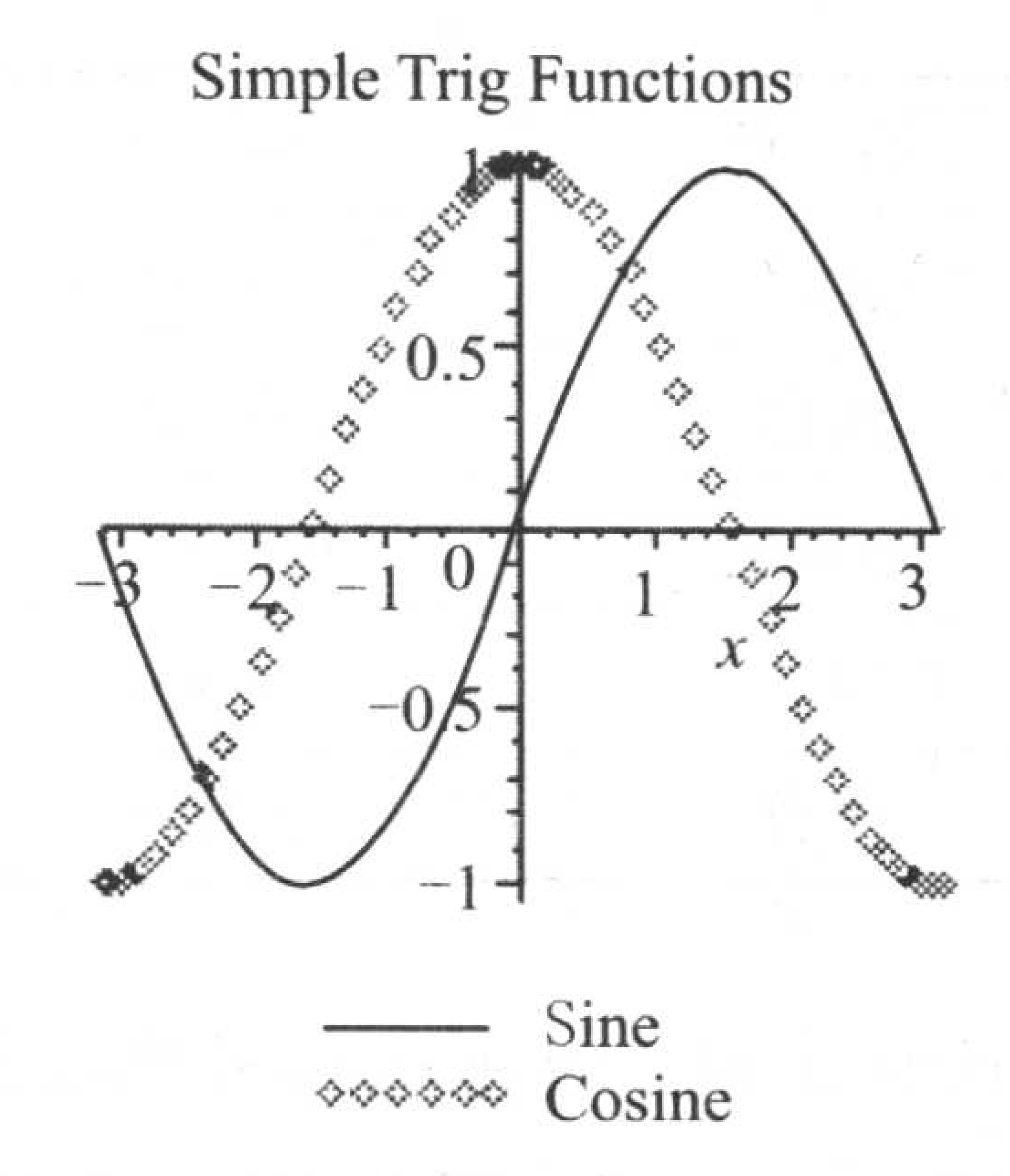


图 6-18

【例 13】 设置曲线的样式、比例,不显示坐标轴(图 6-19)。

```
> plot(cos(x),x=0..2*Pi, style=point, symbol=cross, axes=none,
  scaling=constrained);
```

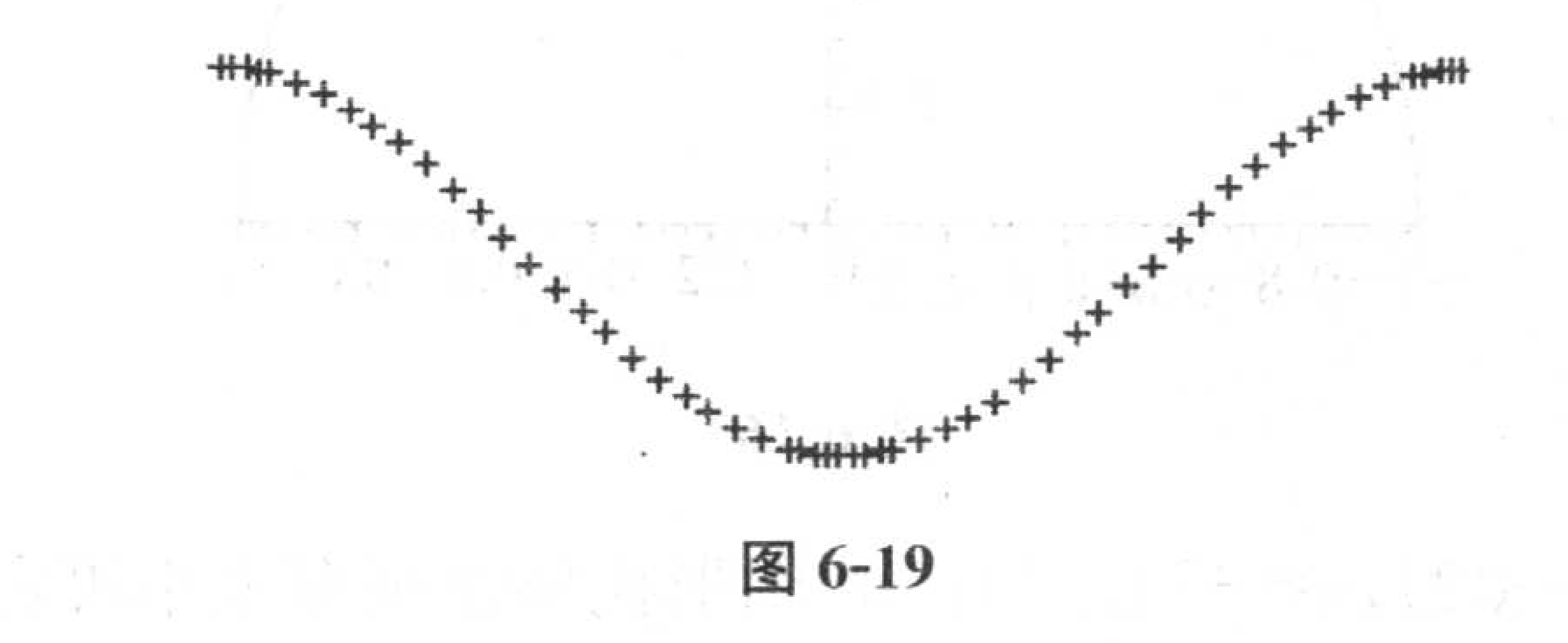


图 6-19

6.1.3 参数方程作图

如果函数用参数方式表示,也能用 plot 画一条参数曲线或一组参数曲线。

```
plot([x(t),y(t),t=a..b],选项)
plot([[x(t),y(t),t=a..b],[u(t),v(t),t=c..d]],选项)
```

【例 14】 画参数曲线 $\begin{cases} x = 1 + \cos t \\ y = \sin t \end{cases}$, $t \in [0,2\pi]$(图 6-20)。

```
> plot([1+cos(t),sin(t),t=0..2*Pi]);
```

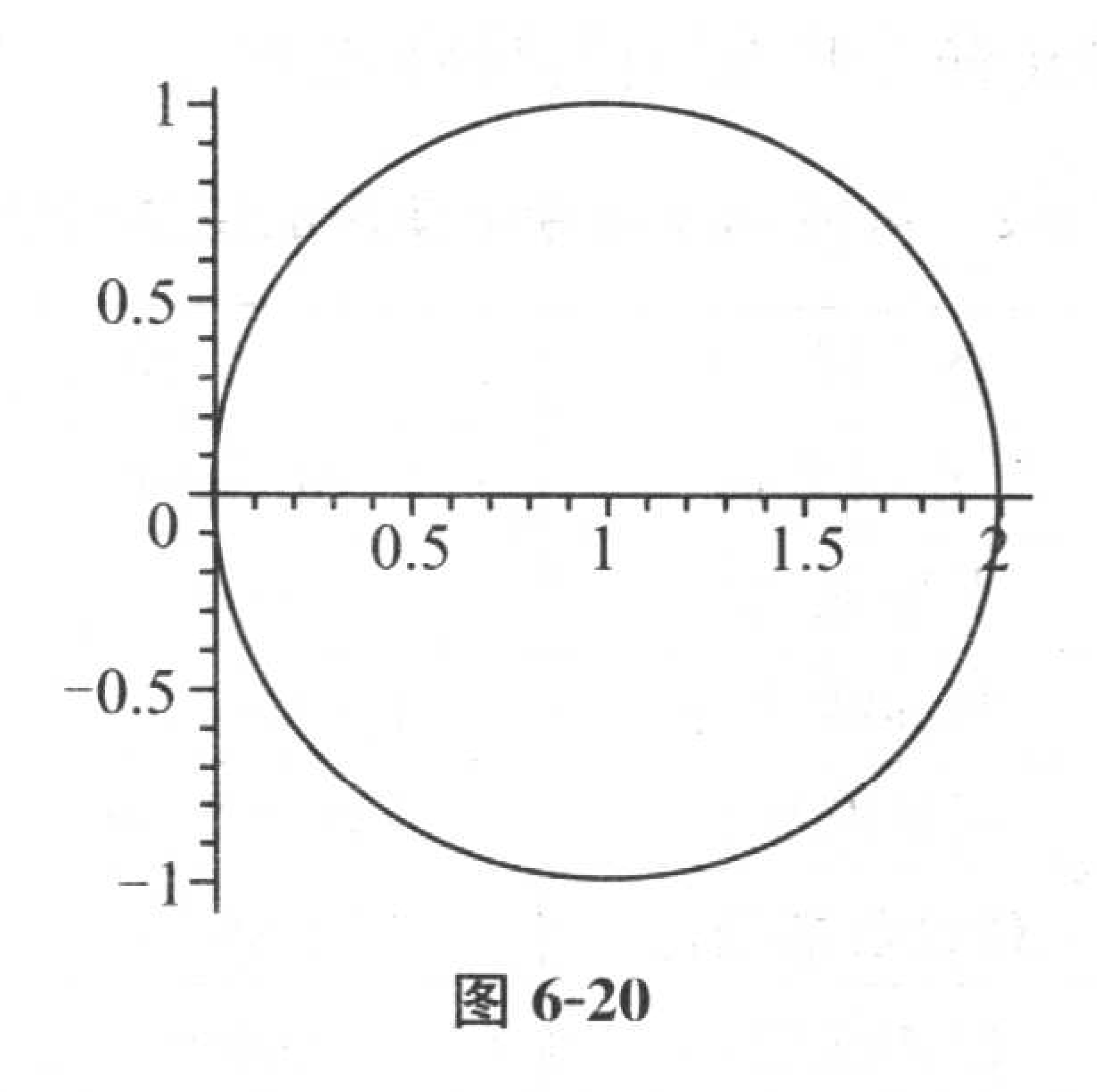


图 6-20

【例 15】　同时画出参数曲线 $\begin{cases} x_1 = \cos t \\ y_1 = \sin t \end{cases}$，$t \in [-\pi, 0]$ 和 $\begin{cases} x_2 = 3\cos t \\ y_2 = 2\sin t \end{cases}$，$t \in [0, \pi]$（图 6-21）。

```
> plot([[cos(t),sin(t),t=-Pi..0],[3*cos(t),2*sin(t),t=0..Pi]]);
```

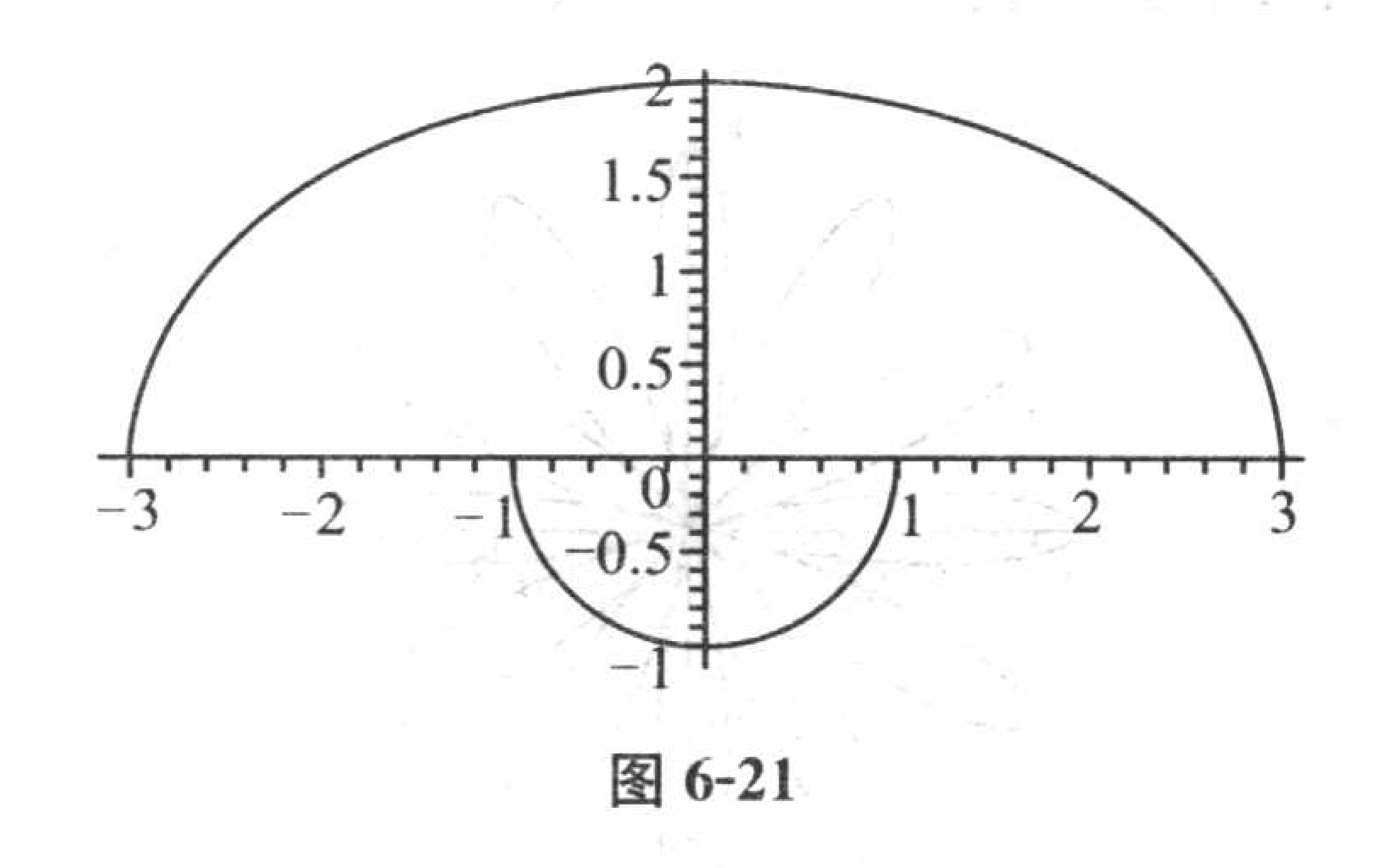


图 6-21

6.1.4　特殊坐标系下作图

通常用 plot 画直角坐标下的函数图像，即系统取默认值 coords=cartesian 为笛卡儿坐标。通过设置 coords 选项，plot 也可以画出特殊坐标下的函数图像。例如，画极坐标下函数 $r = r(t)$，$a \leqslant t \leqslant b$ 的图形可用命令：

```
plot(r(t),t=a..b,coords=polar)
plot([r(t),t,t=a..b],coords=polar)
```

在 6.3 节中，还将给出 plots 函数包中画特殊坐标系下的函数图形命令，例如：

```
polarplot(r(t),t=a..b)
```

表 6-2 列出了可在 plot 命令中使用的特殊坐标系。

表 6-2　可在 plot 命令中使用的特殊坐标系

选　　项	坐　标　系	选　　项	坐　标　系
bipolar	双极坐标	cardioid	心形坐标
cassinian	卡尼亚坐标	elliptic	椭圆坐标
hyperbolic	双曲线坐标	invcassinian	反卡尼亚坐标
invelliptic	反椭圆坐标	logarithmic	对数坐标
logcosh	对数余割坐标	maxwell	麦克斯韦坐标
parabolic	抛物线坐标	polar	极坐标
rose	玫瑰坐标	tangent	切线坐标
cartesian	笛卡儿坐标		

【例 16】　特殊坐标系下的函数图像(图 6-22～6-26)。

```
> plot([cos(6 * x),x,x=0..2 * Pi],coords=polar);
```

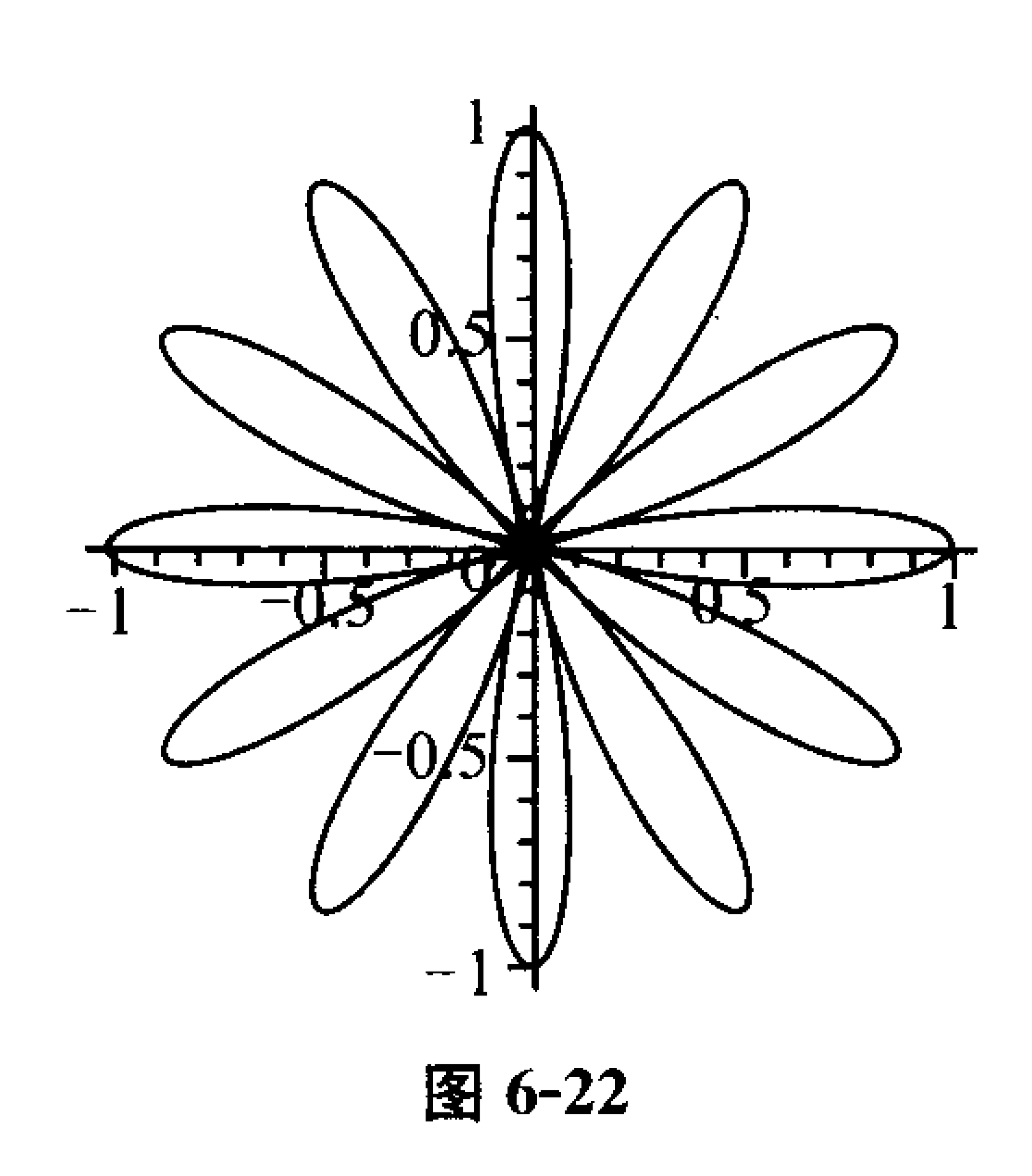


图 6-22

```
> plots[polarplot](3 * sin(t)+3.5 * cos(8 * t) * cos(6 * t),t=0..2 * Pi);
```

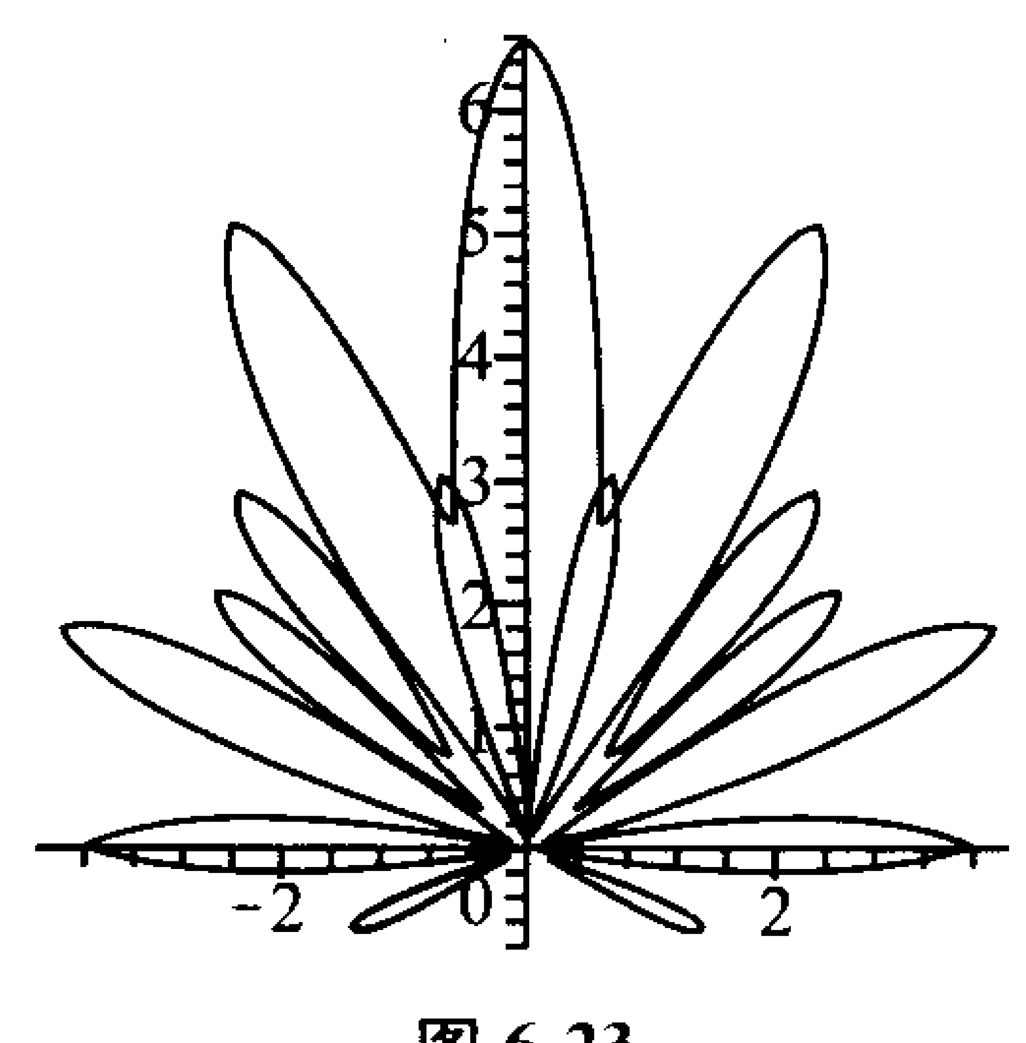


图 6-23

> plot(cos(6 * t),t=－Pi..Pi,coords=bipolar)；＃ 双极坐标

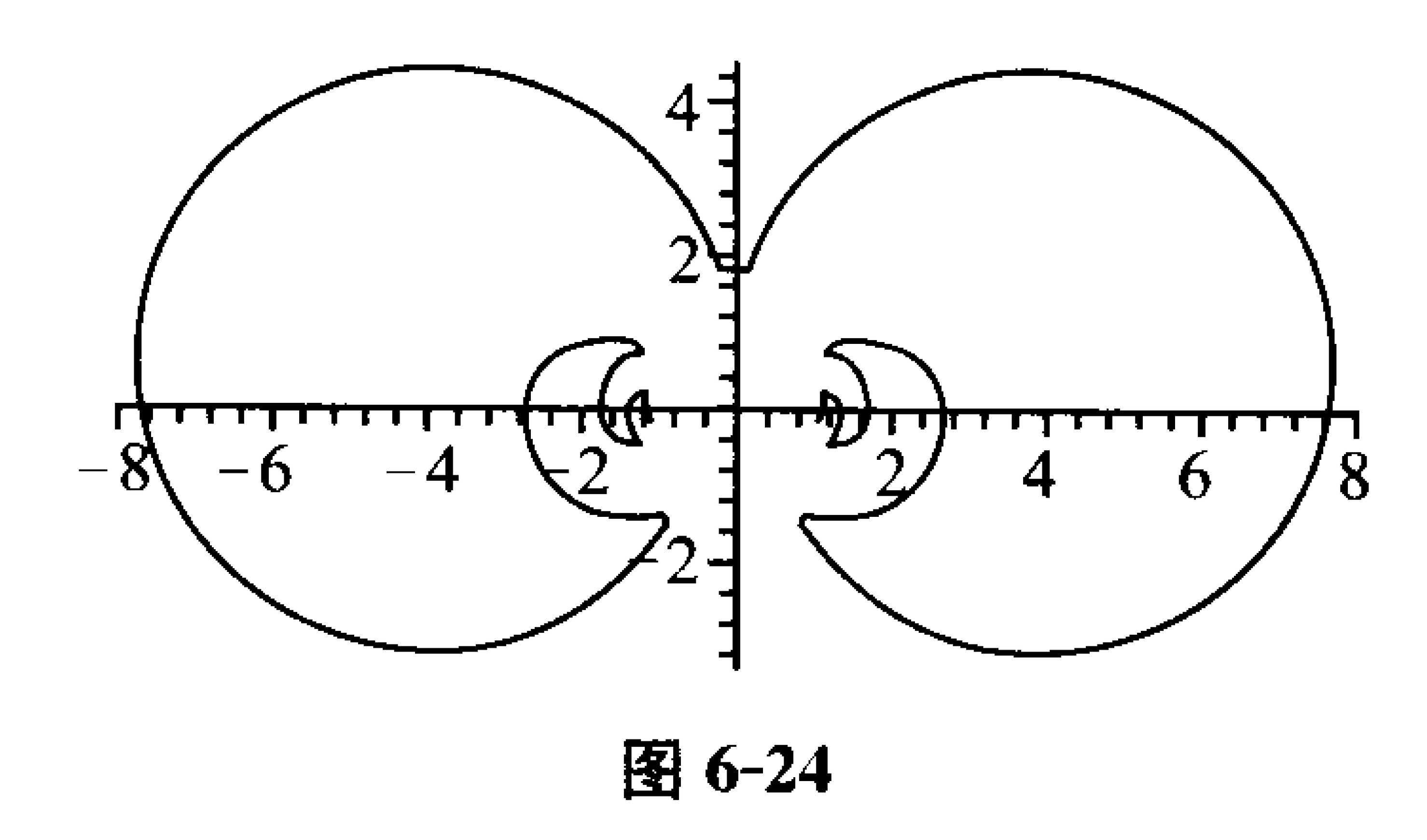


图 6-24

> plot(cos(6 * t),t=－Pi..Pi,coords=elliptic)；＃ 椭圆坐标

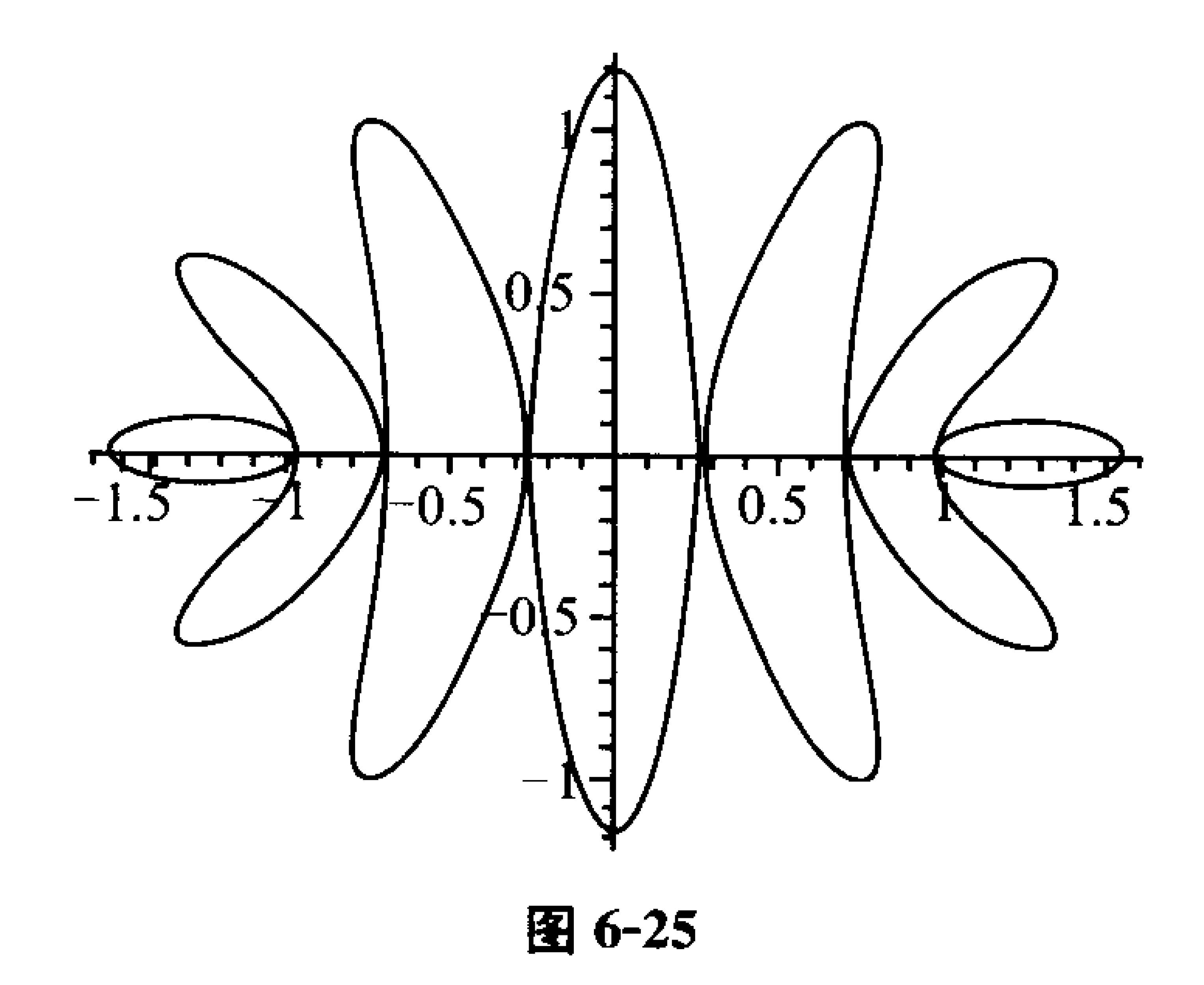


图 6-25

> plot(cos(6 * t),t=－Pi..Pi,coords=maxwell)；＃ 麦克斯韦坐标

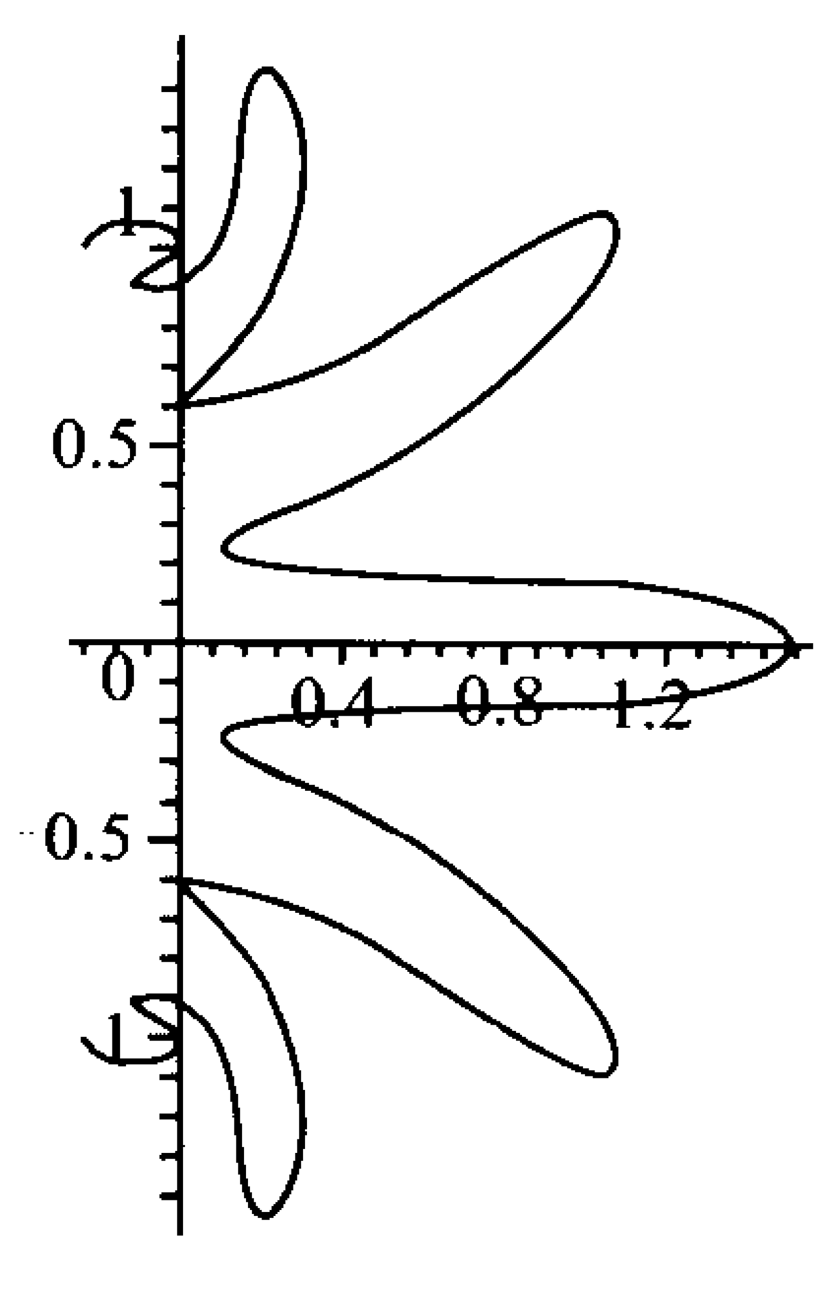


图 6-26

6.2　三维函数作图命令(plot3d)

6.2.1　三维函数作图

画二元函数 $f(x,y)$的三维图像可用 plot3d 命令：

plot3d(f(x,y),x=a..b,y=c..d,选项)

与 plot 函数相似,plot3d 的选项可有可无,项数可多可少。plot3d 的部分选项的名称和功能的细节请看 6.2.3 节。

【例 17】　画出函数 $f(x,y)=\dfrac{1}{1+\sin^4 x+\cos^4 y}$, $-\pi\leqslant x,y\leqslant\pi$ 的图像(图 6-27)。

```
> plot3d(1/(1+sin(x)^4+cos(y)^4),x=-Pi..Pi,y=-Pi..Pi);
```

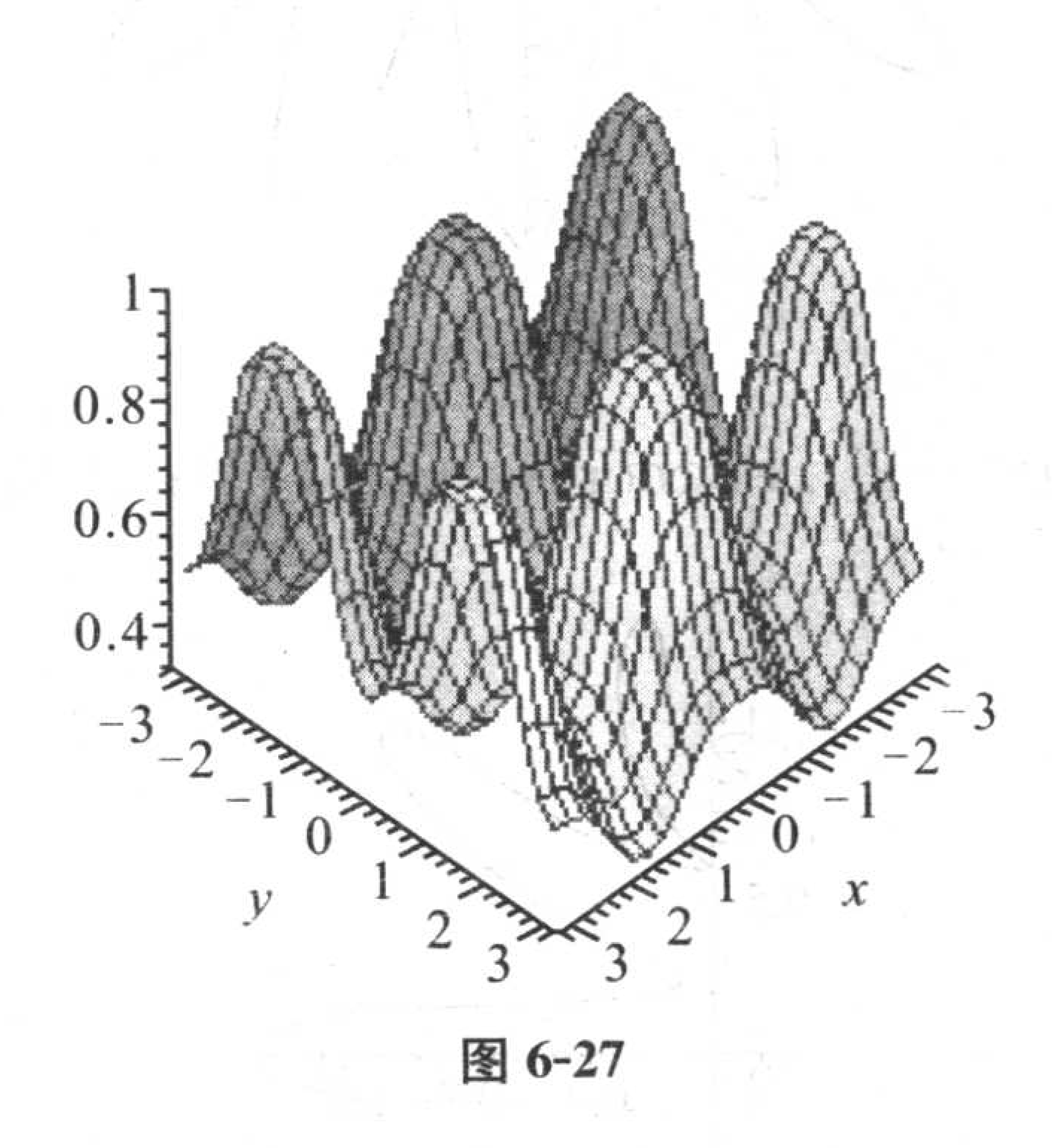


图 6-27

单击 plot3d 图形,此时的上下文工具栏不同于 plot 图形的上下文工具栏(图 6-28)。

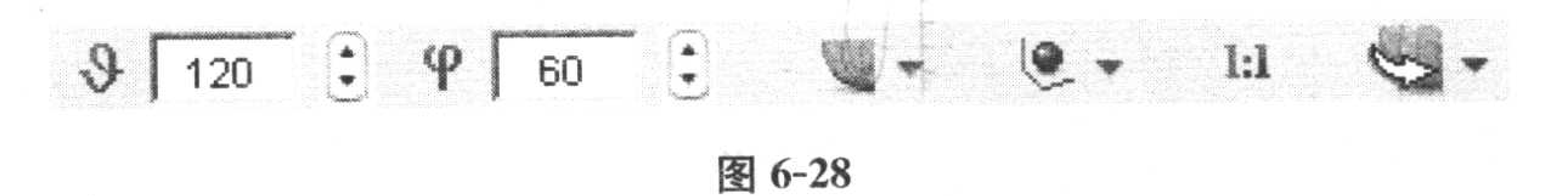


图 6-28

工具栏中各按钮的作用见图 6-29。

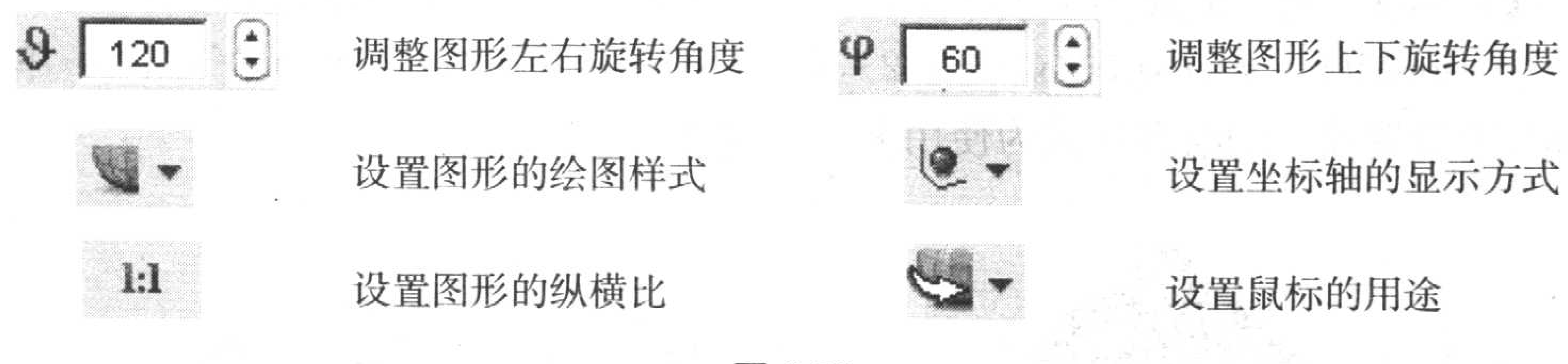


图 6-29

【例 18】 画出双曲抛物面(马鞍面)$z=x^2-y^2$，$-1\leqslant x,y\leqslant 1$(图 6-30)。

```
> plot3d(x^2-y^2,x=-1..1,y=-1..1);
```

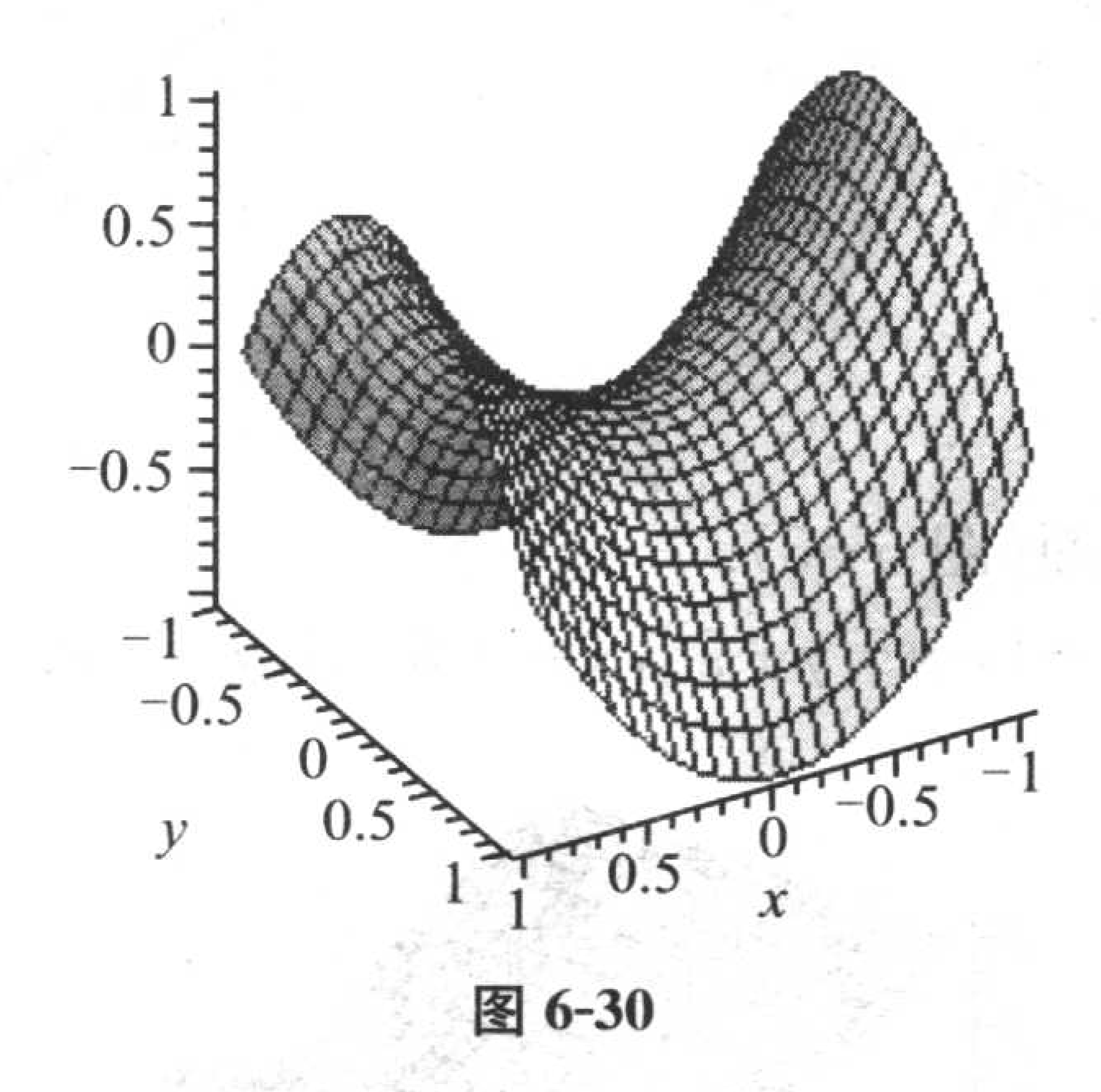


图 6-30

【例 19】 画出函数 $B(x,y)=\dfrac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}$，$-2\leqslant x,y\leqslant 2$ 的图形(图 6-31)。

```
> plot3d(Beta,-2..2,-2..2);
```

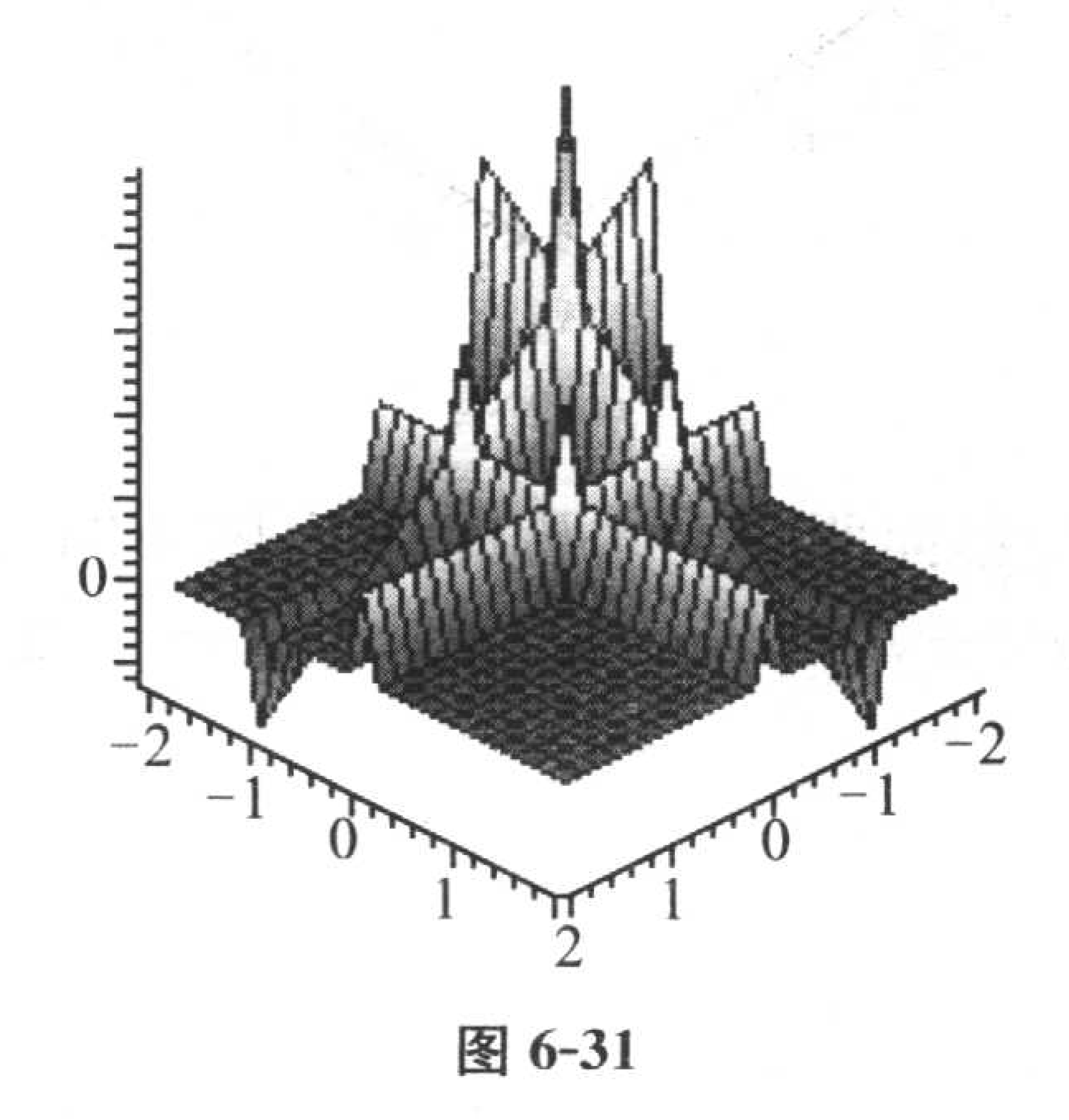


图 6-31

【例 20】 画出函数 $f(x,y)=xye^{-x^2-y^2}$，$-2\leqslant x,y\leqslant 2$ 的图形(图 6-32)。

> plot3d(x * y * exp(-x^2-y^2),x=-2..2,y=-2..2,color=x)； # 单击图形,再设置图形的绘图样式为使用多边形面片

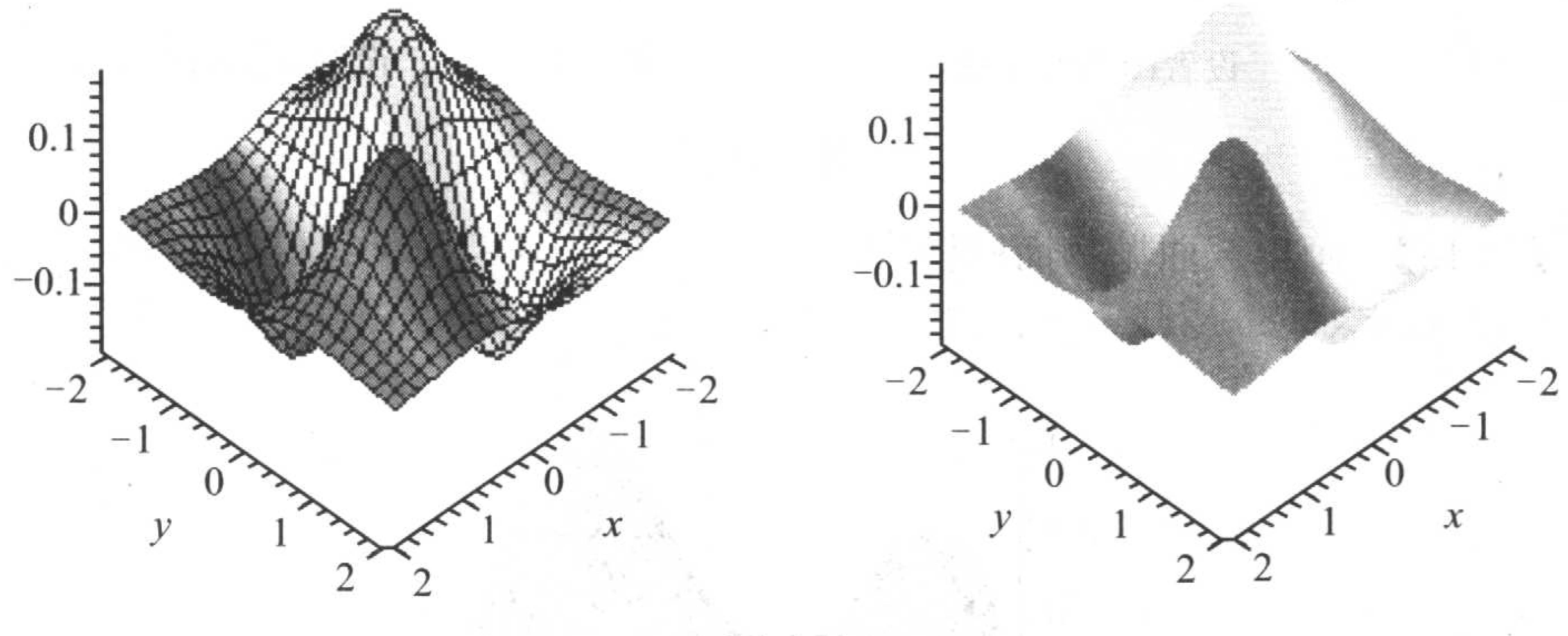


图 6-32

【例 21】 两相交平面(图 6-33)。

> plot3d([x+2 * y-1, x-y+2],x=-3..3,y=-3..3)；

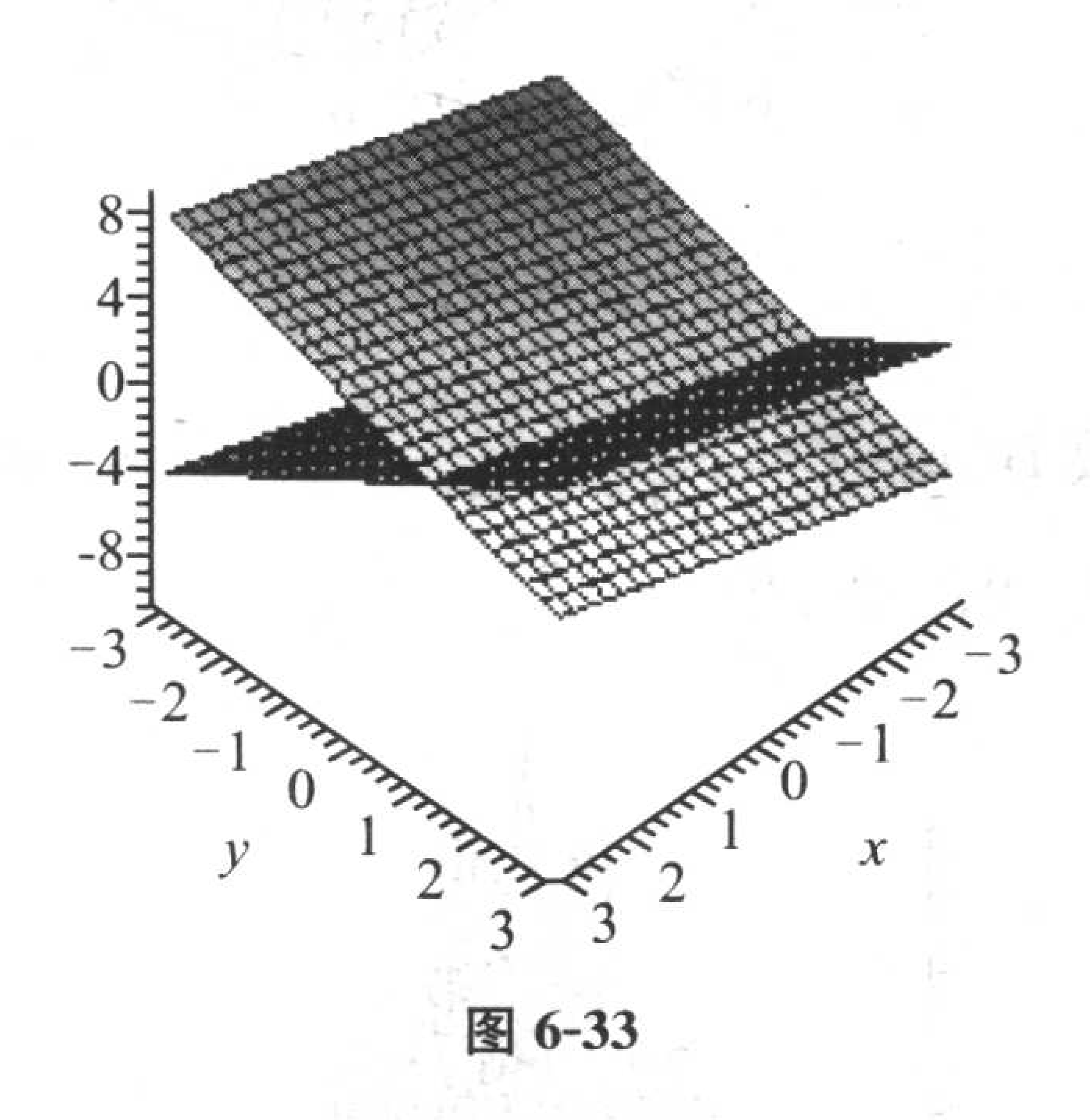


图 6-33

【例 22】 画出球面与锥面所围立体及其表面(图 6-34)。

> plot3d([sqrt(8-x^2-y^2),sqrt(x^2+y^2)],x=-2..2,y=-2..2)；

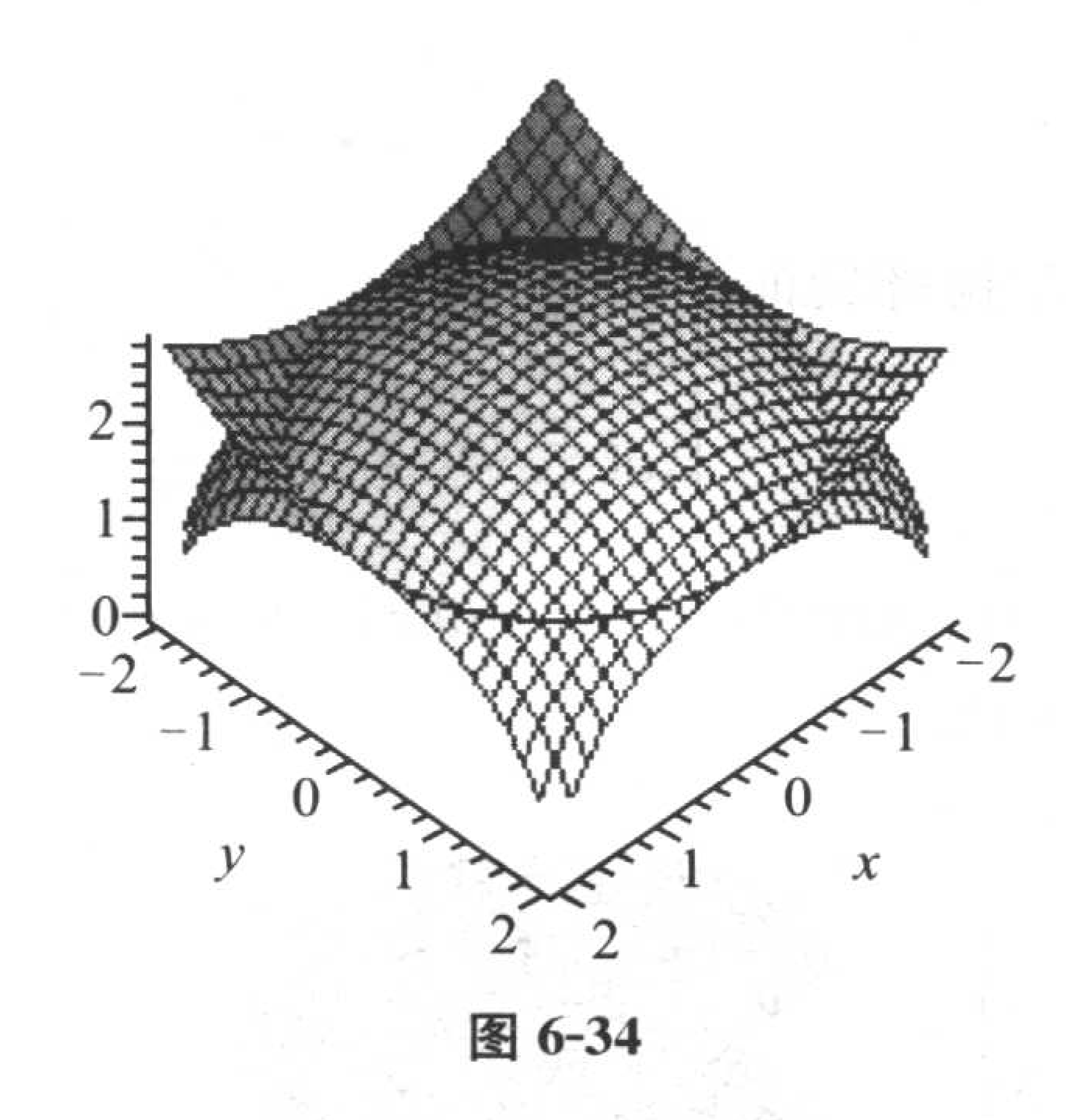


图 6-34

6.2.2　三维参数方程作图

用 plot3d 命令画三维参数曲面 $(x(s,t),y(s,t),z(s,t))$ 的一般形式为：

```
plot3d([x(s,t),y(s,t),z(s,t)],s=a..b,t=c..d)
plot3d([f,g,h],a..b,c..d)
```

其中 x、y、z 是关于 s 和 t 的表达式，f、g、h 是有两个参数的过程。

【例 23】 画出参数曲面 $\begin{cases} u(x,y) = x\sin x\cos y \\ v(x,y) = x\cos x\cos y, \\ w(x,y) = x\sin y \end{cases}$ $\begin{matrix} 0 \leqslant x \leqslant 2\pi \\ 0 \leqslant y \leqslant \pi \end{matrix}$（图 6-35）。

```
> plot3d([x * sin(x) * cos(y),x * cos(x) * cos(y),x * sin(y)],
  x=0..2 * Pi,y=0..Pi,labels=['u','v','w']);
```

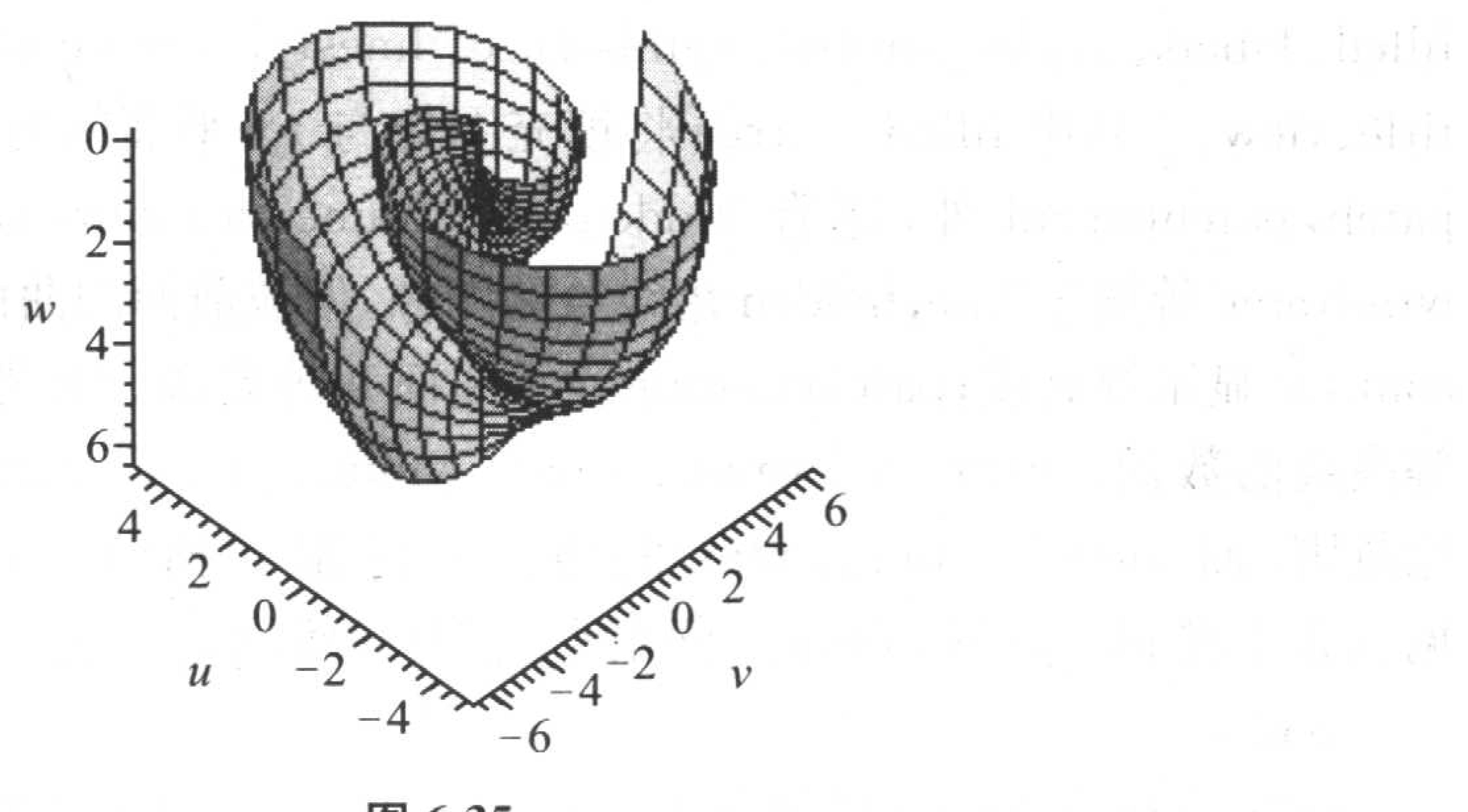


图 6-35

【例 24】 画出阿基米德螺旋面 $\begin{cases} u(x,y) = -\dfrac{t}{\sqrt{2}}\cos s \\ v(x,y) = -\dfrac{t}{\sqrt{2}}\sin s \\ w(x,y) = \dfrac{t}{\sqrt{2}} + \dfrac{s}{2\pi} \end{cases}$，$\begin{matrix} 0 \leqslant t \leqslant 2 \\ 0 \leqslant s \leqslant 5\pi \end{matrix}$（图 6-36）。

```
> plot3d([-t*cos(s)/sqrt(2),-t*sin(s)/sqrt(2),t/sqrt(2)+s/2/Pi],
  t=0..2,s=0..5*Pi);
```

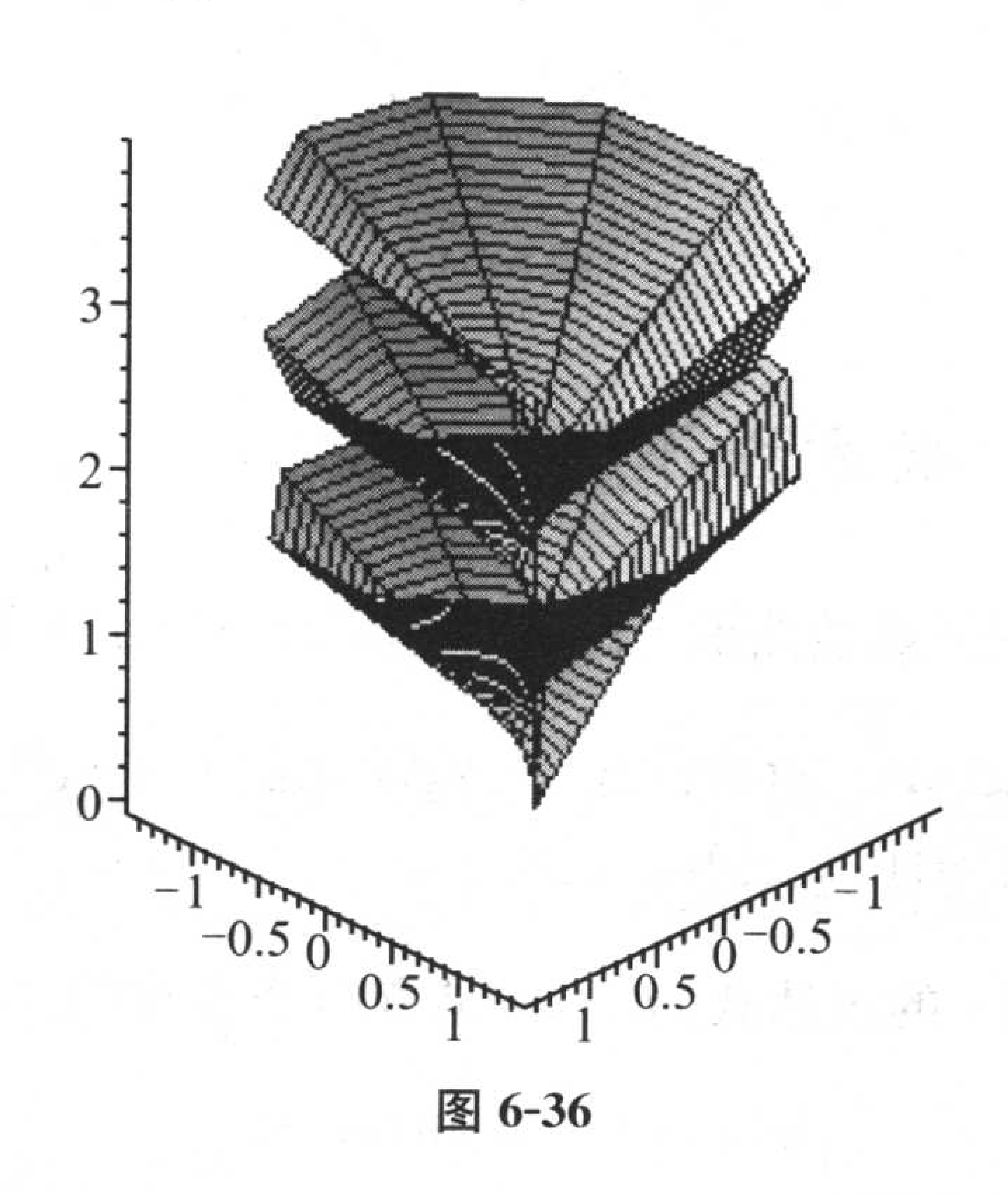


图 6-36

6.2.3 plot3d 选项

plot3d 命令部分选项的名称和功能与 plot 的选项相似，例如：axes、coords、filled、labels、style、symbol、symbolsize、linestyle、numpoints、scaling、thickness、title、view。其中 filled = true 将填充图像与 xy 平面间的区域。style 除了 point，patch，patchnogrid 外，还有 hidden，wireframe，contour，patchcontour 这些取值。wireframe 等同于 line；hidden 用线绘制图像，但被前面的曲面遮挡住的线而不显示；contour 显示等高线；patchcontour 显示曲面和等高线。另外 contours=n 可以控制等高线的数量。view = [xmin..xmax, ymin..ymax, zmin..zmax]控制显示的图像范围，而 view = min..max 的格式只控制 z 轴的显示范围。coords 取值请见 6.2.4 或 plot3d[coords]。下面列出部分三维图形选项：

grid

grid=[m,n]将 xy 区间分成 $m*n$ 个方格。m 与 n 越大，图像越清晰。

gridstyle

控制网格形状，取值 rectangular 或 triangular。当 style＝patch、line 或 hidden 时有效。

color 或 colour

控制图像的色彩。预定义的色彩和 color＝COLOR(HUE/RGB,…)精确定义的色彩同样有效。这样三维图像内外表面用单色显示。而三维图形经常根据 3 个分量值绘制随图像变化的色彩，plot3d 支持 color 等于函数或过程的着色法。如果是一个值，将以 HUE 来理解；如果是 3 个值组成的序列，将以 RGB 的 3 个分量来理解。另外由于参数表示的问题，这时 color 用函数还是过程来表示，必须跟前面图像的表示一致。

shading

另一个关于表明如何着色的选项，取值 yz, xy, z, zgrayscale, zhue 或 non。但 color 的设置更优先。

light

light＝[phi,theta,r,g,b]从球坐标 phi,theta 方向增加一个 r,g,b 表示的色彩的直射光源。r,g,b 必须介于 0 与 1 之间。

lightmodel

用一个预定义的光源照射图像，这些光源包括 none、light1、light2、light3、light4。

ambientlight

ambientlight＝[r,g,b]设置一个 RGB 表示的漫射光。

orientation

orientation＝[theta,phi]设置图像观察者的入射角度，相当于在什么位置观察图像。用球坐标系下的两个角度 theta 和 phi 来表示，默认这两个角度都是 45 度。

projection

projection＝r 设置透视效果，这里 r 取 0 到 1 之间的实数。1 代表正交投影，0 代表广角透视，而 fisheye、normal、orthogonal 分别对应 r 取 0、0.5、1。默认值为 orthogonal。

【例 25】 设置 axes、grid 选项(图 6-37)。

```
> plot3d([s*cos(t)*(2+cos(s+t)),s*sin(t)*(2+cos(s+t)),s*sin(s+
  t)], s=0..3*Pi,t=0..2*Pi,axes=none,grid=[50,50]);
```

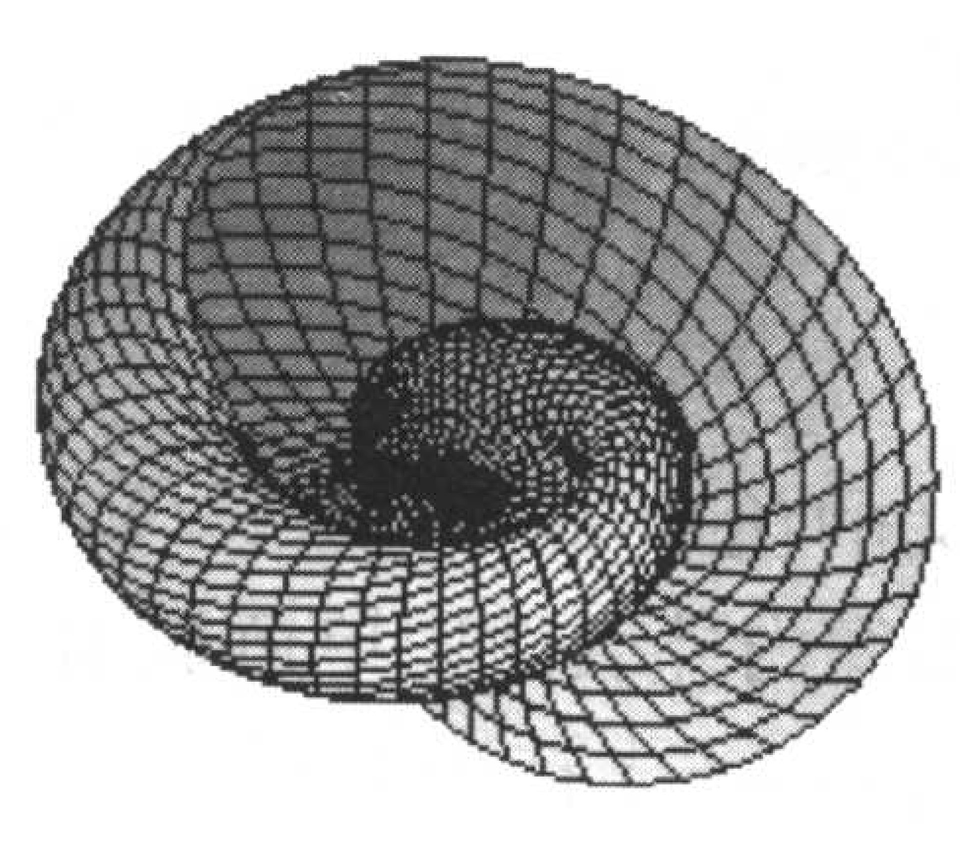


图 6-37

【例 26】 画出用过程定义的参数曲面。观察不同选项 style 下的图形(图 6-38)。

```
> f:=(x,y)->sin(x)*cos(y): g:=(x,y)->sin(x)*sin(y):
  h:=(x,y)->cos(x)/1.5: plot3d([f,g,h],Pi/4..3*Pi/4,0..2*Pi);
```

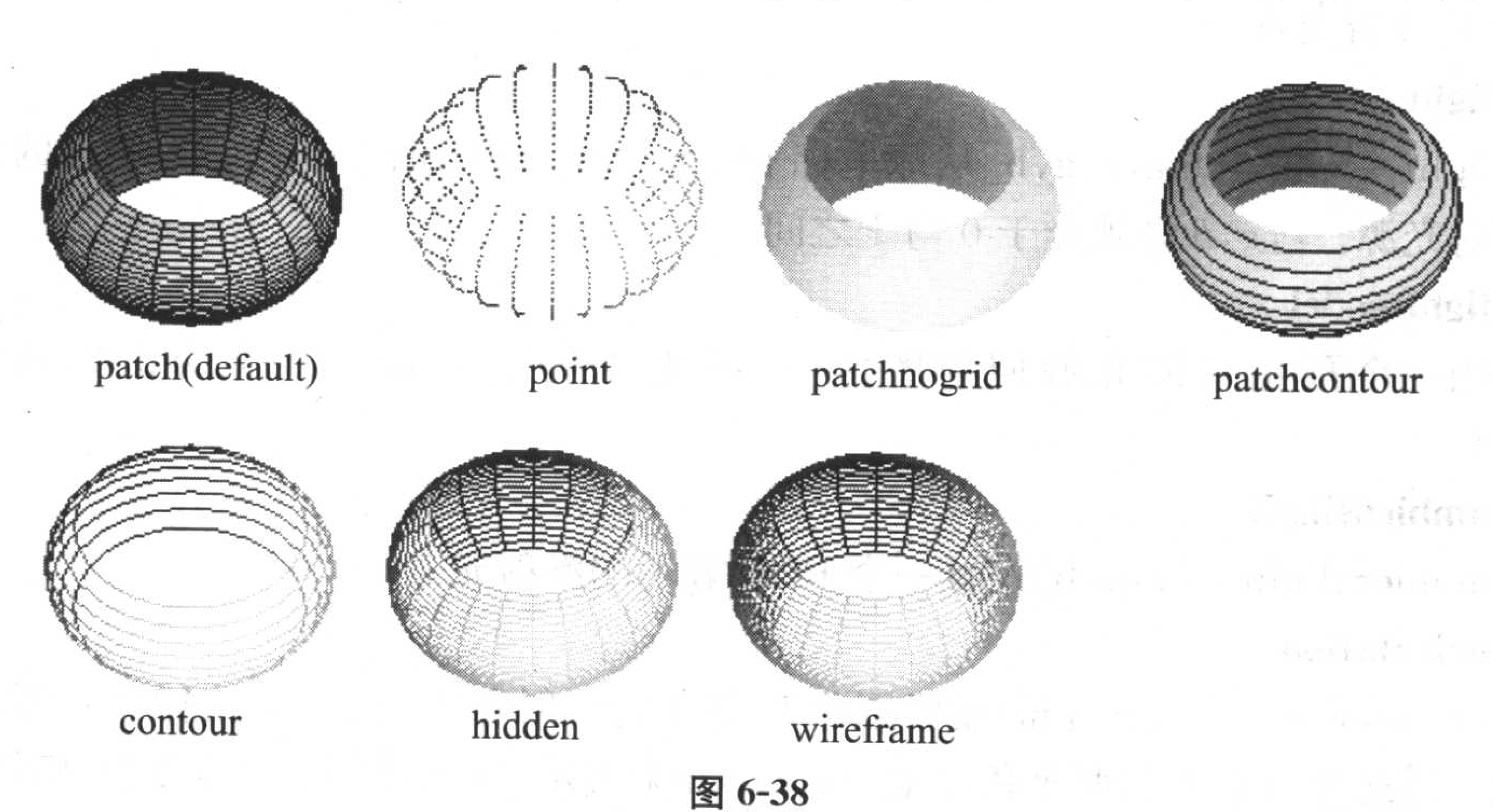


图 6-38

【例 27】 观察比较不同选项的效果(图 6-39)。

```
> plot3d(-y*exp(-x^2-y^2),x=-2..2,y=-2..2);
```

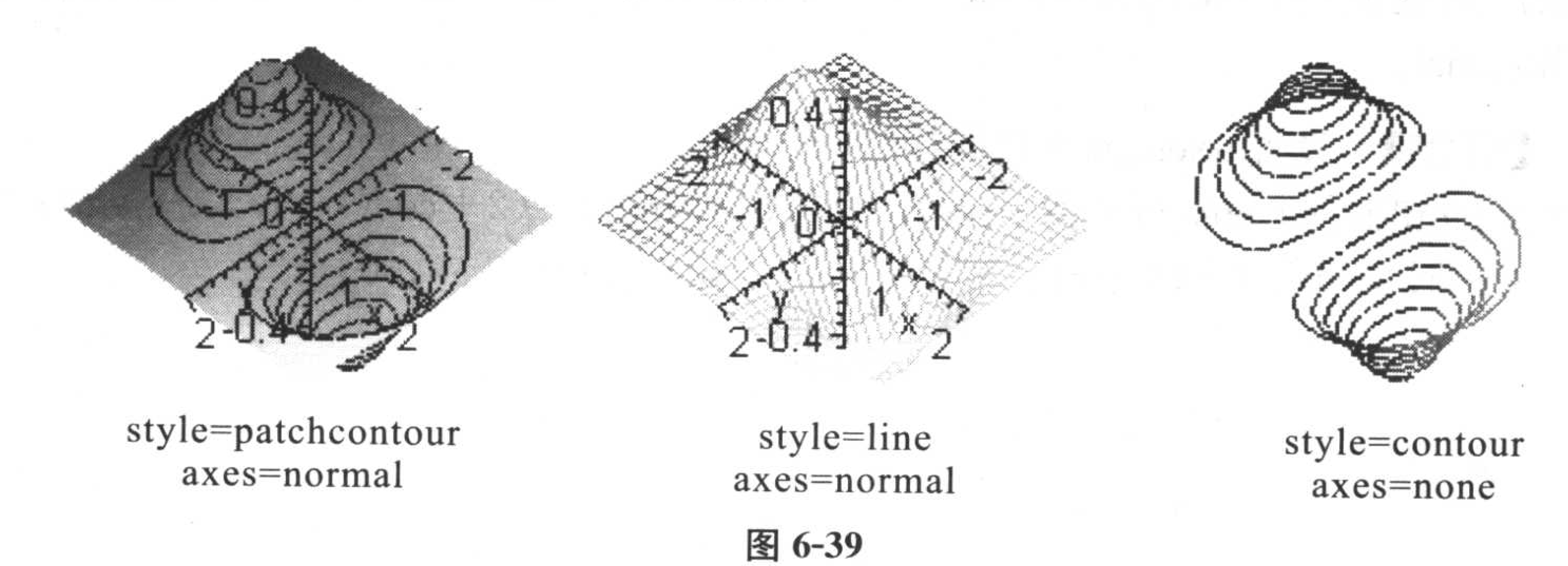


图 6-39

6.2.4　特殊坐标系下的 plot3d 作图

在 plot3d 作图中，coords 的选项值有 29 个。我们熟悉的有 cylindrical（柱坐标）、spherical（球坐标）。除此之外还有 bipolarcylindrical、bispherical、cardioidal、cardioidcylindrical、casscylindrical、confocalellip、confocalparab、conical、ellcylindrical、ellipsoidal、hypercylindrical、invcasscylindrical、invellcylindrical、invoblspheroidal、invprospheroidal、logcoshcylindrical、logcylindrical、maxwellcylindrical、oblatespheroidal、paraboloidal、paracylindrical、prolatespheroidal、rosecylindrical、sixsphere、tangentcylindrical、tangentsphere、toroidal。

【例 28】　用柱坐标作图（图 6-40）。

```
> plot3d(height,angle=0..2*Pi,height=-5..5, coords=cylindrical,
  title=`CONE`);
```

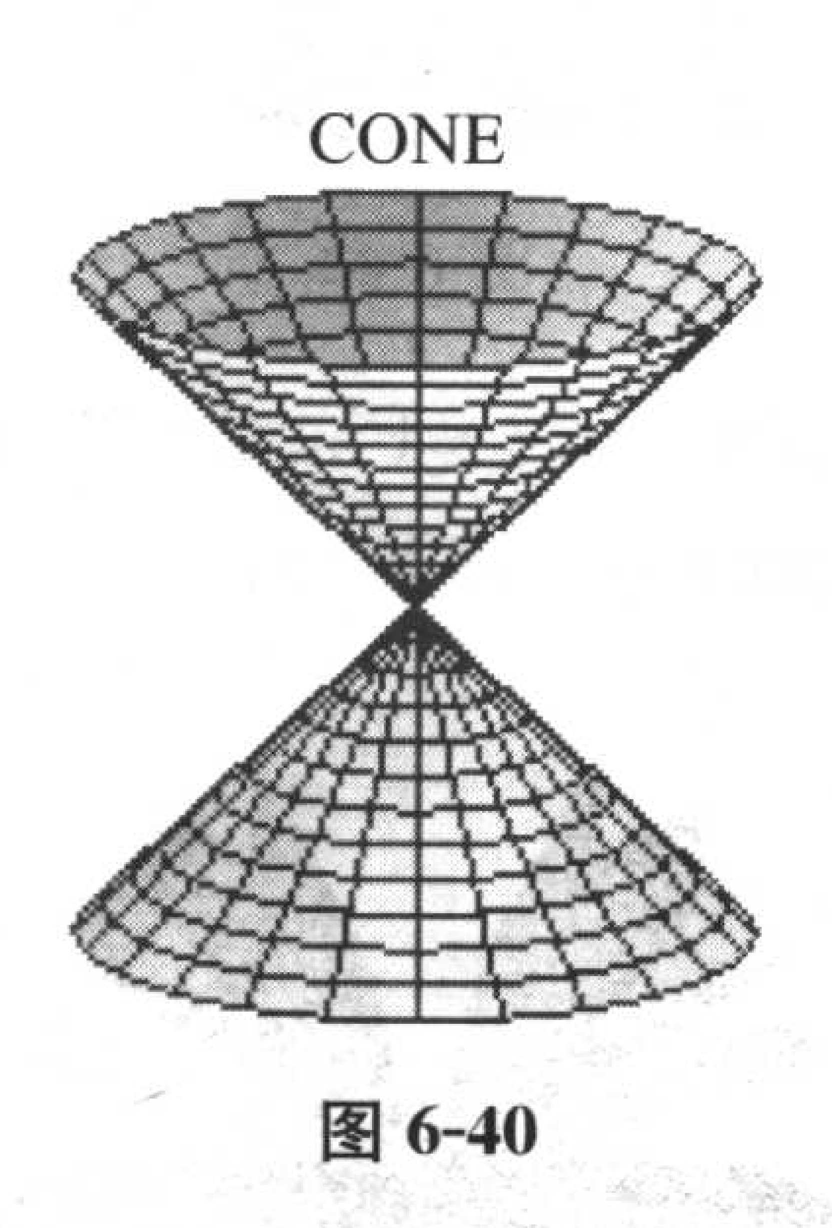


图 6-40

【例 29】　在柱坐标和球坐标下作函数 $f(r,\theta)=r\cos\theta$ 的图形（图 6-41）。

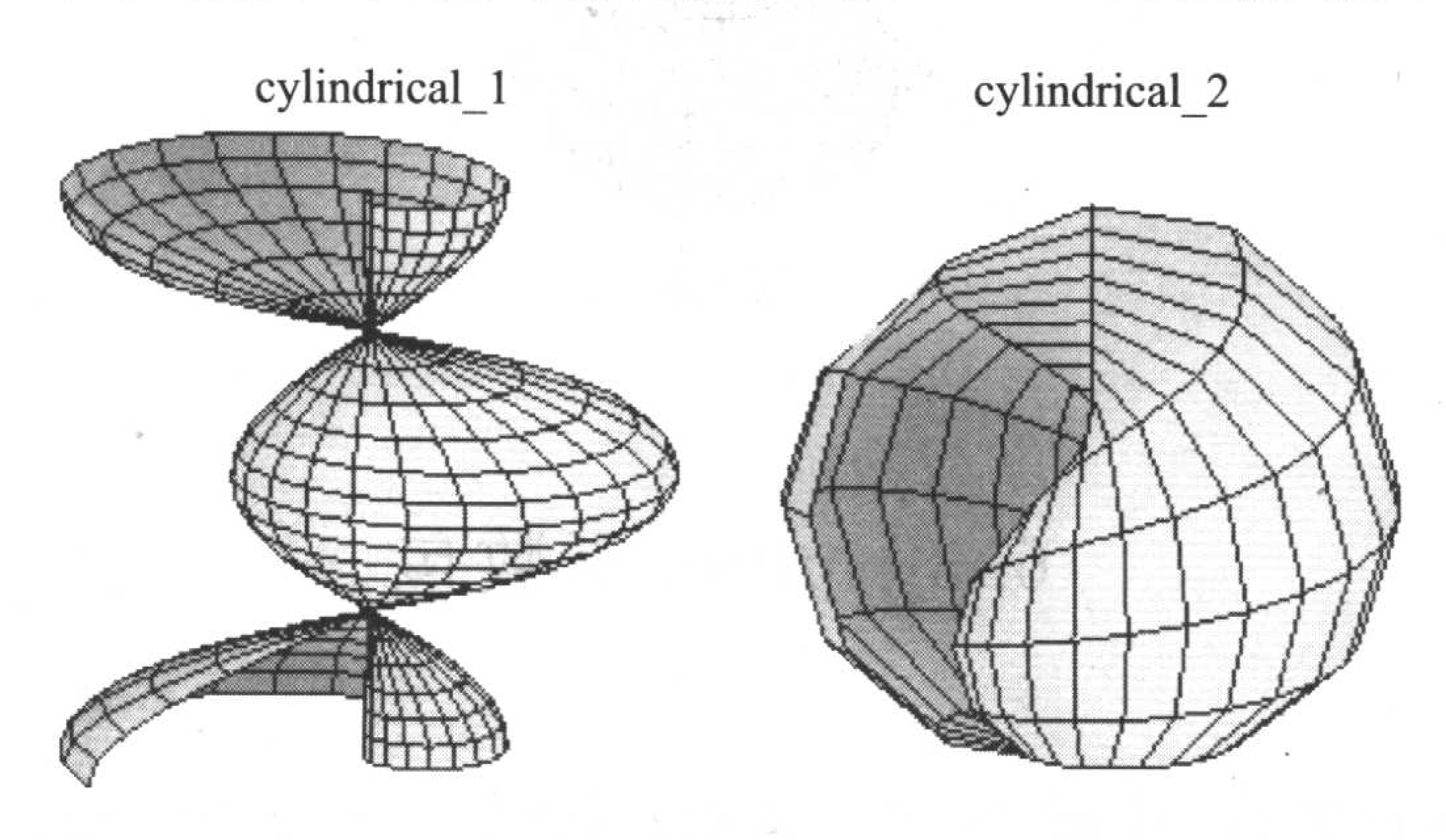


图 6-41

```
> plot3d(r * cos(theta),r=0..6,theta=0..2 * Pi, coords=cylindrical,
  title=`cylindrical_1`);
  plot3d(r * cos(theta),r=0..6,theta=0..2 * Pi, coords=spherical,
  title=`cylindrical_2`);
```

【例 30】 用球坐标作图(图 6-42,6-43)。

```
> plot3d(1,t=0..2 * Pi,p=0..Pi,coords=spherical); # 单位球
```

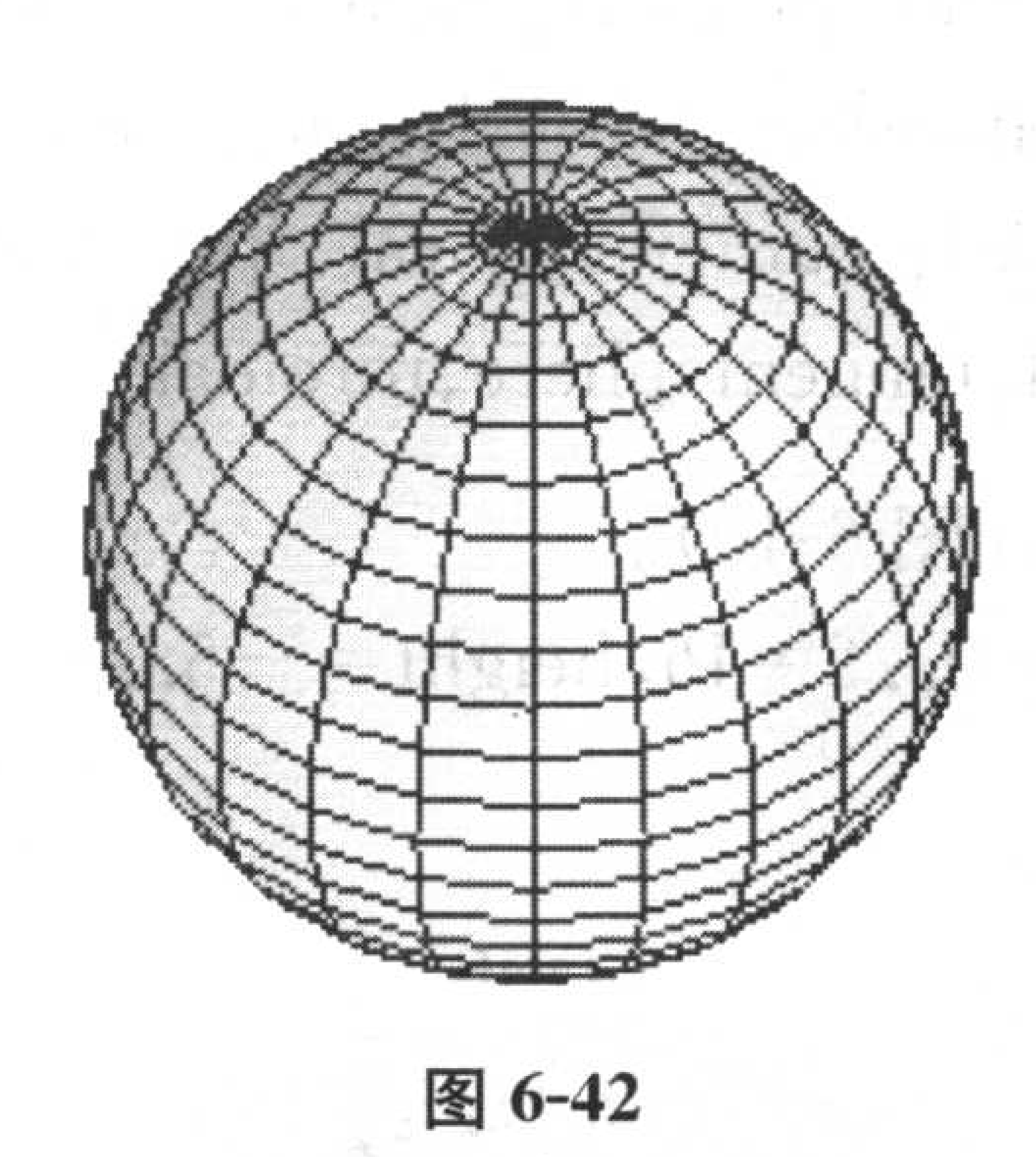


图 6-42

```
> plot3d(cos(x^2) * sin(y),x=-2 * Pi..2 * Pi,y=-Pi..Pi,
  coords=spherical, numpoints=10000);
```

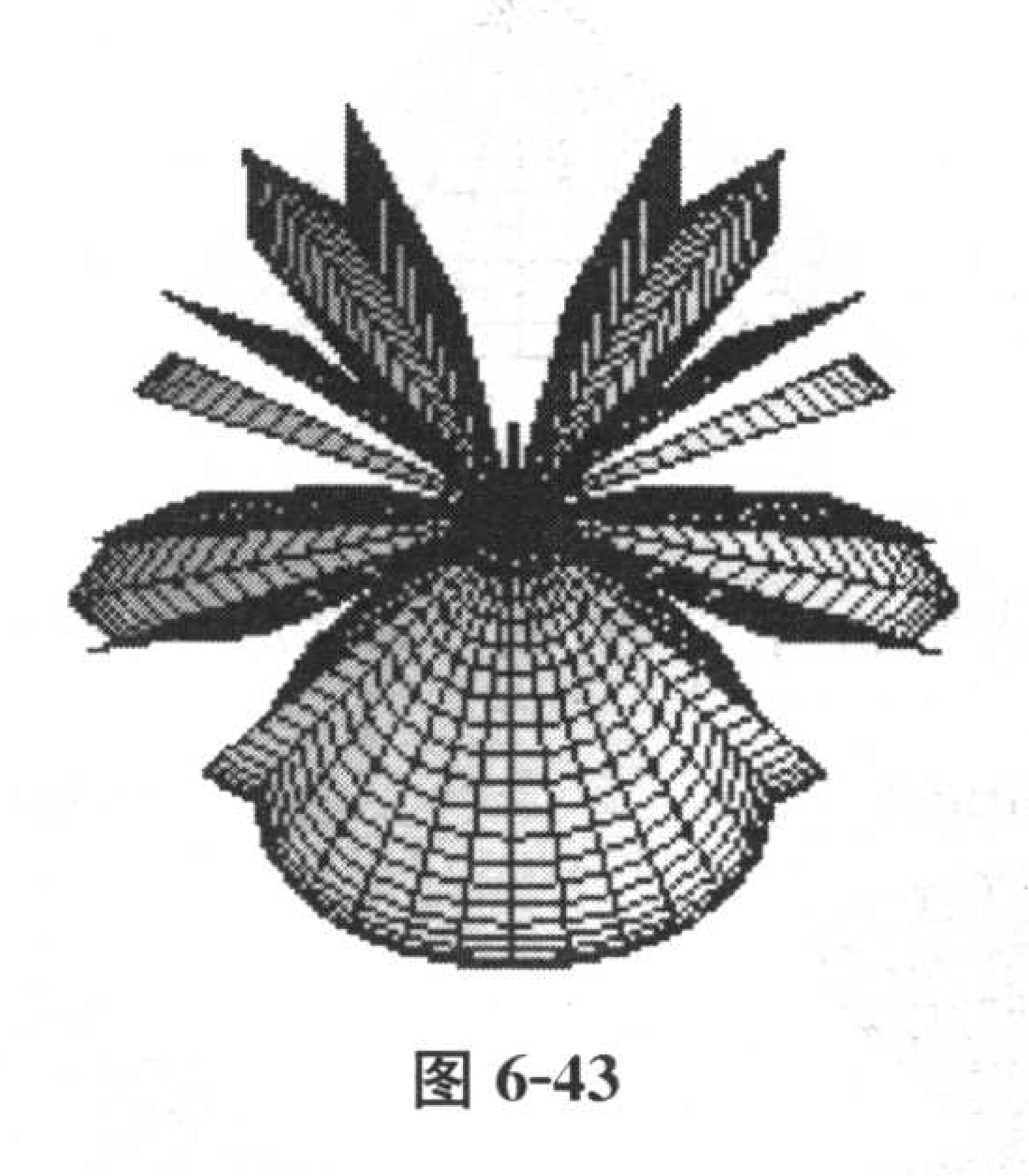


图 6-43

6.3 plots 函数包

在 plots 包中有 56 个画图命令和选项设置命令,在本节中只能列出部分命令。动画演示命令 animate、animate3d 和 animatecurve 在 6.4 节中介绍,绘制微分方程

数值解的函数图像的命令 odeplot 在第 5 章中介绍。

```
> with(plots);
```

[animate, animate3d, animatecurve, arrow, changecoords, complexplot, complexplot3d, conformal, conformal3d, contourplot, contourplot3d, coordplot, coordplot3d, densityplot, display, display3d, fieldplot, fieldplot3d, gradplot, gradplot3d, graphplot3d, implicitplot, implicitplot3d, inequal, interactive, interactiveparams, listcontplot, listcontplot3d, listdensityplot, listplot, listplot3d, loglogplot, logplot, matrixplot, multiple, odeplot, pareto, plotcompare, pointplot, pointplot3d, polarplot, polygonplot, polygonplot3d, polyhedra_supported, polyhedraplot, replot, rootlocus, semilogplot, setoptions, setoptions3d, spacecurve, sparsematrixplot, surfdata, textplot, textplot3d, tubeplot]

6.3.1　图形的合成和显示

replot(图形,选项)
display(图形列表,选项)

replot 命令最终将被 display 命令所代替,故推荐使用 display 命令。

【例 31】　观察 3 幅 tan(x)的函数图像(图 6-44～6-46)。

```
> plot(tan(x),x=0..Pi);
```

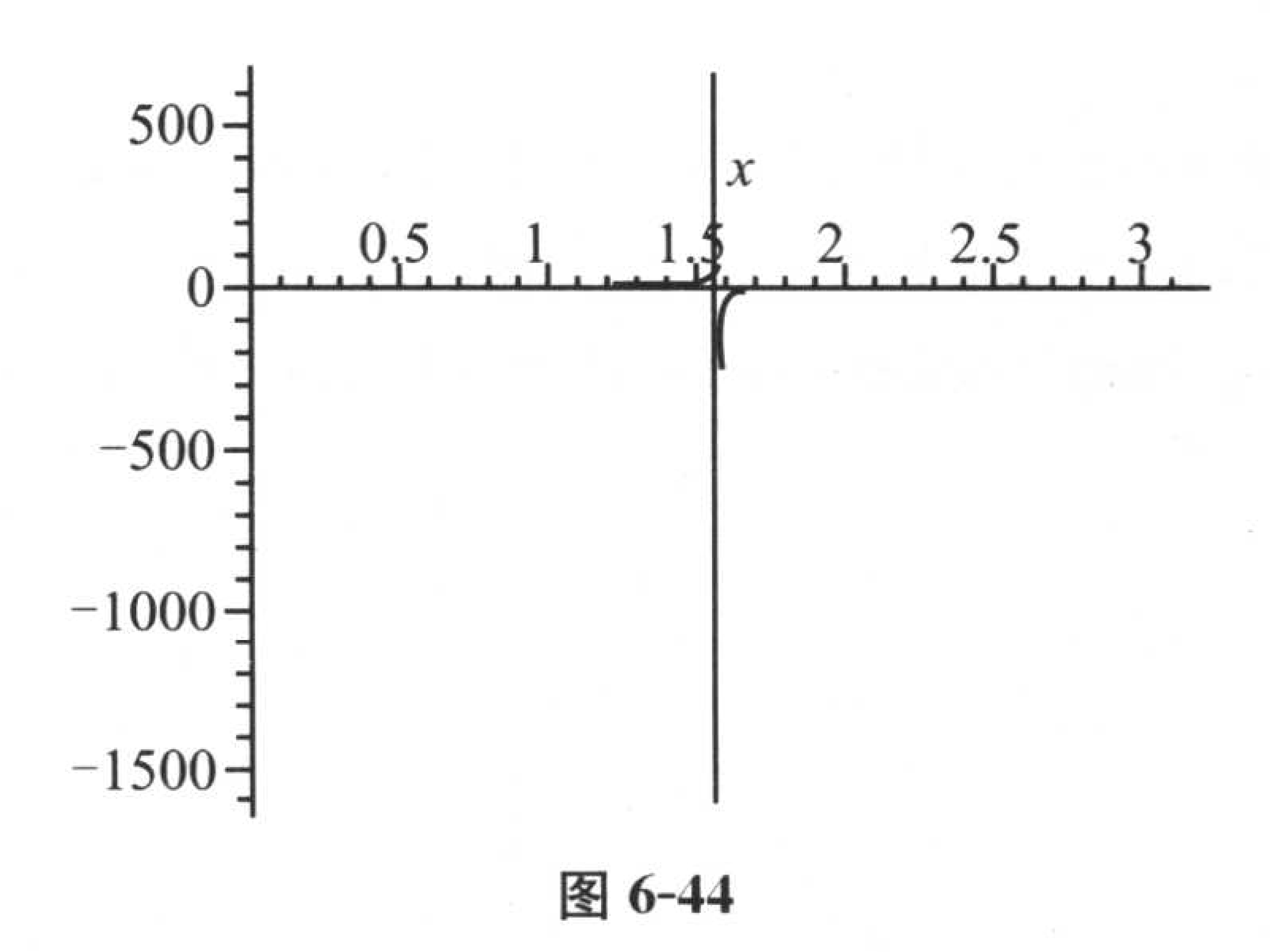


图 6-44

```
> replot(%,view=[0..Pi,-5..5]);
```

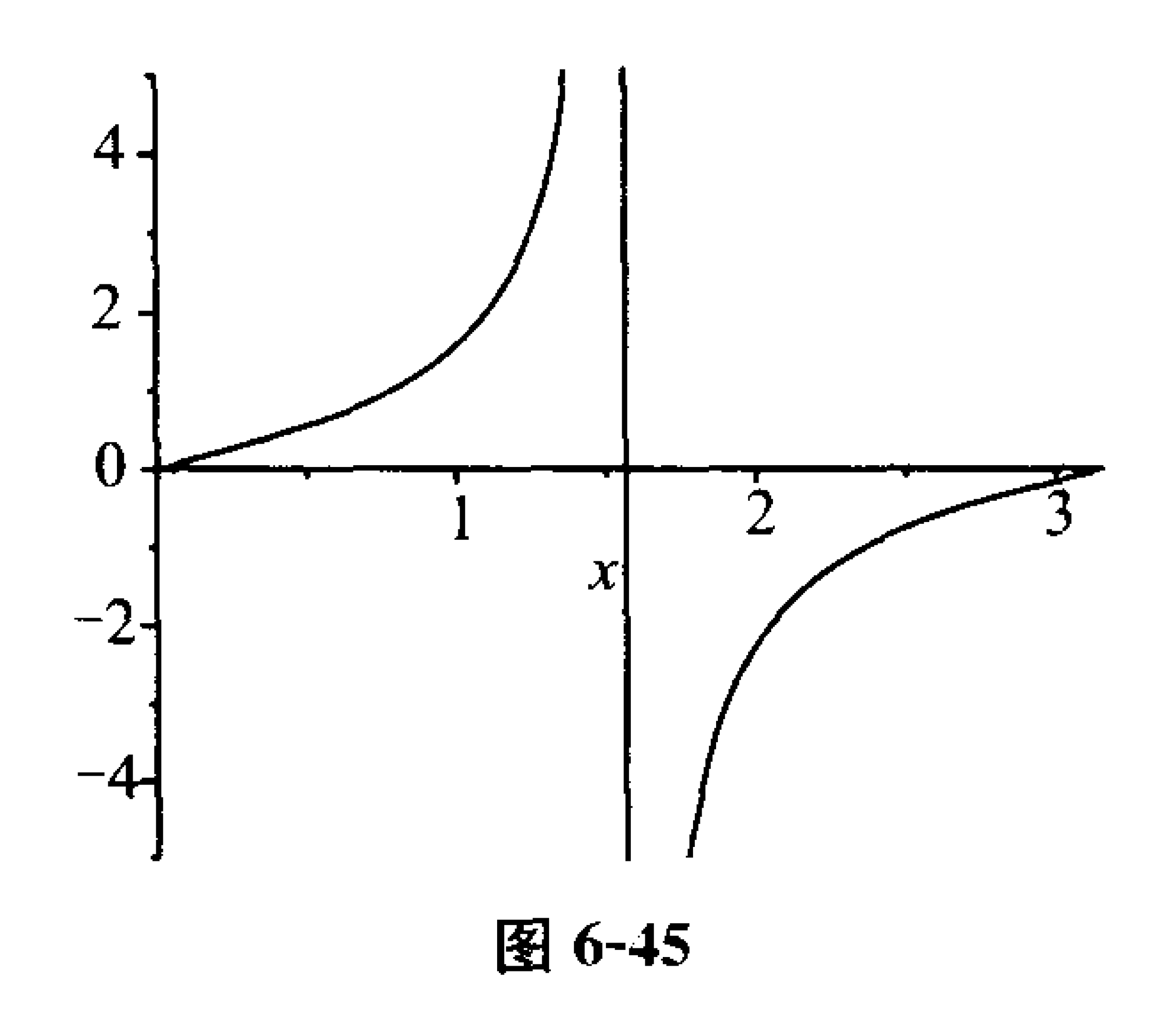


图 6-45

```
> replot(%,style=point,title="y=tan(x)");
```

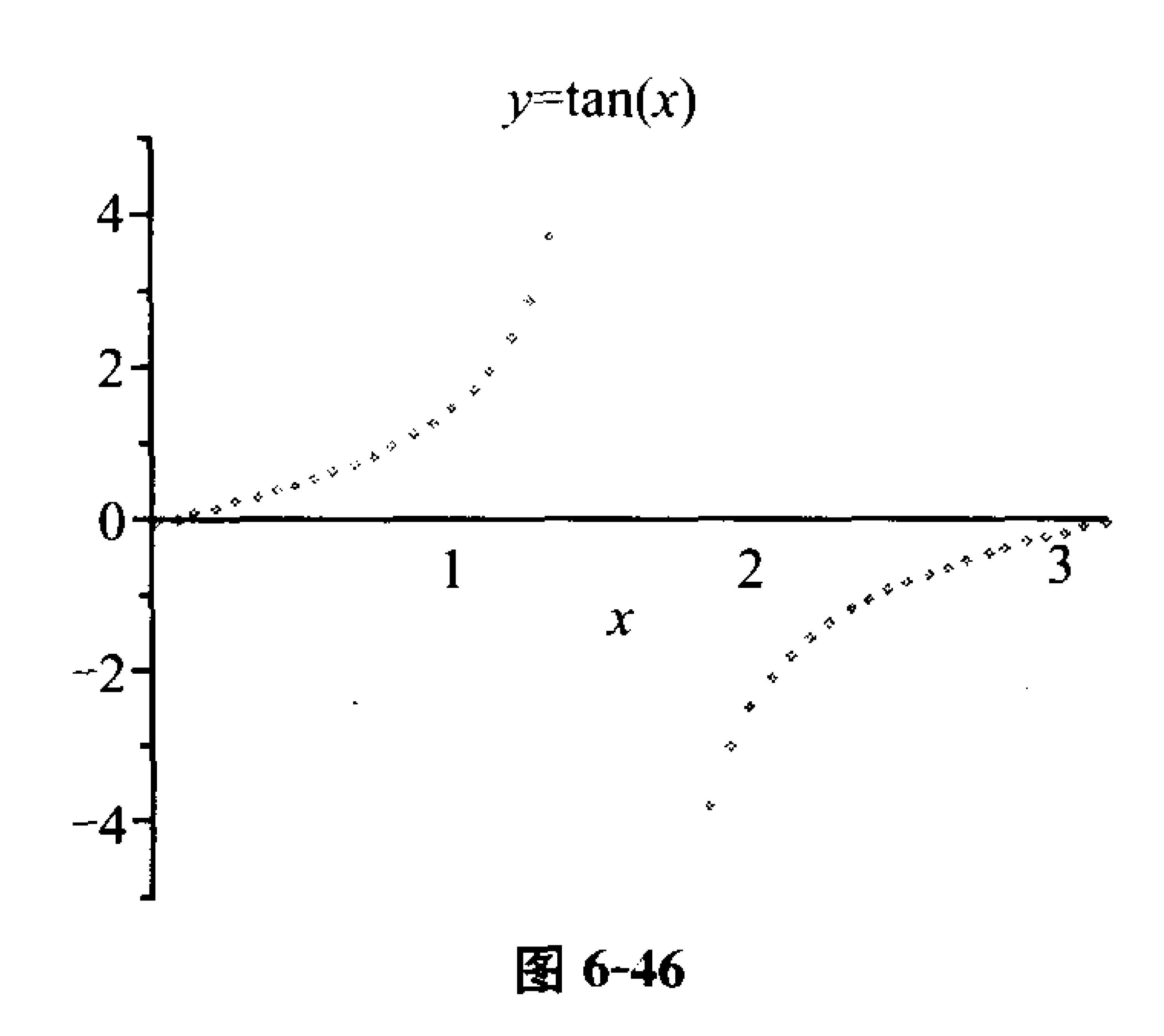


图 6-46

【例 32】 两个图形画在同一个坐标系中 vs 两个图形分别画在不同的坐标系中(图 6-47,6-48)。

```
> F:=plot(cos(x),x=-Pi..Pi,y=-2..2,style=line):
> G:=plot(x^2,x=-2..2,y=0..3,style=point):
> display(F,G,scaling=constrained); # (F,G) 或 {F,G}或 [F,G]
```

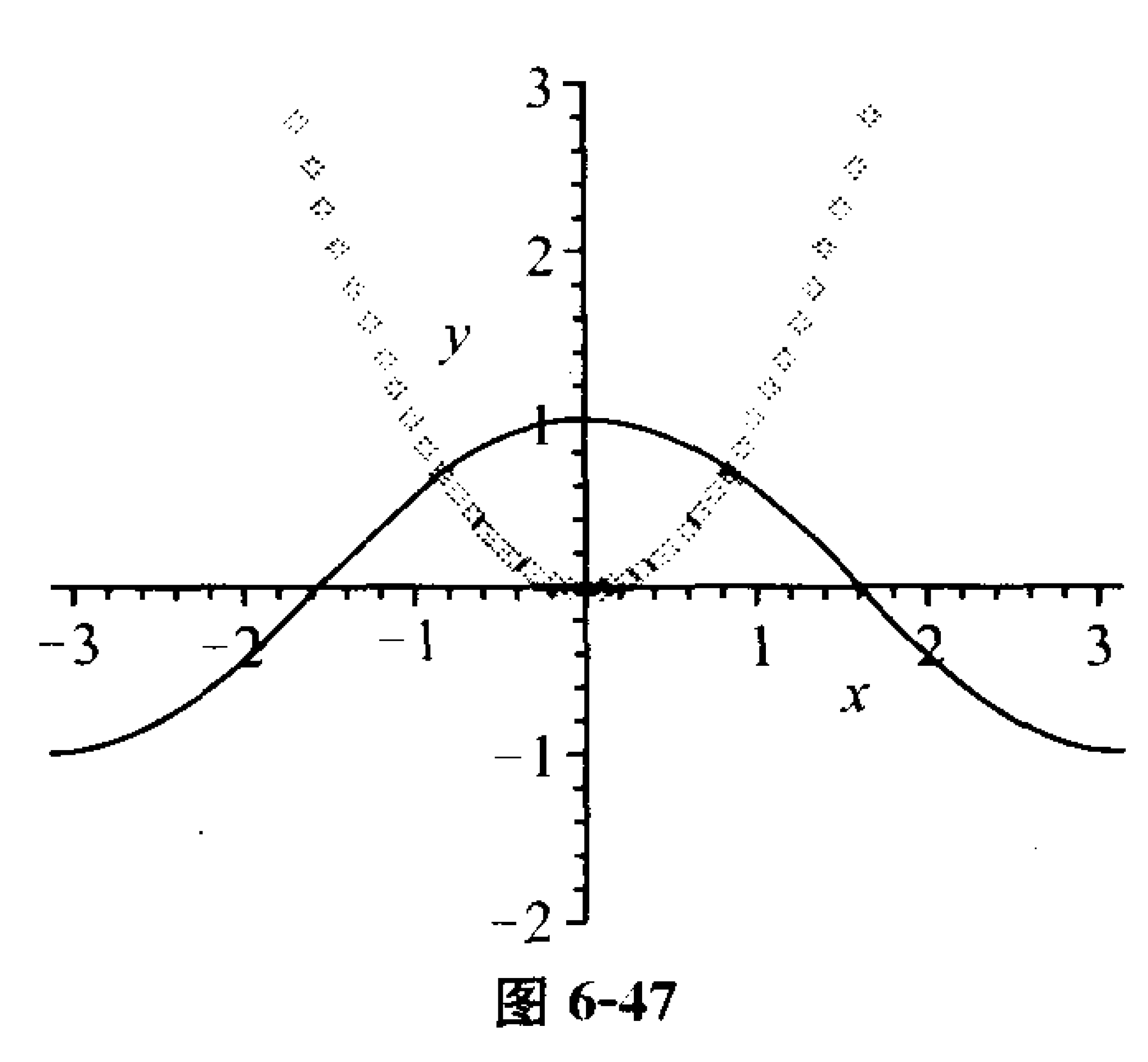


图 6-47

```
> display(array([F,G]),scaling=constrained);
```

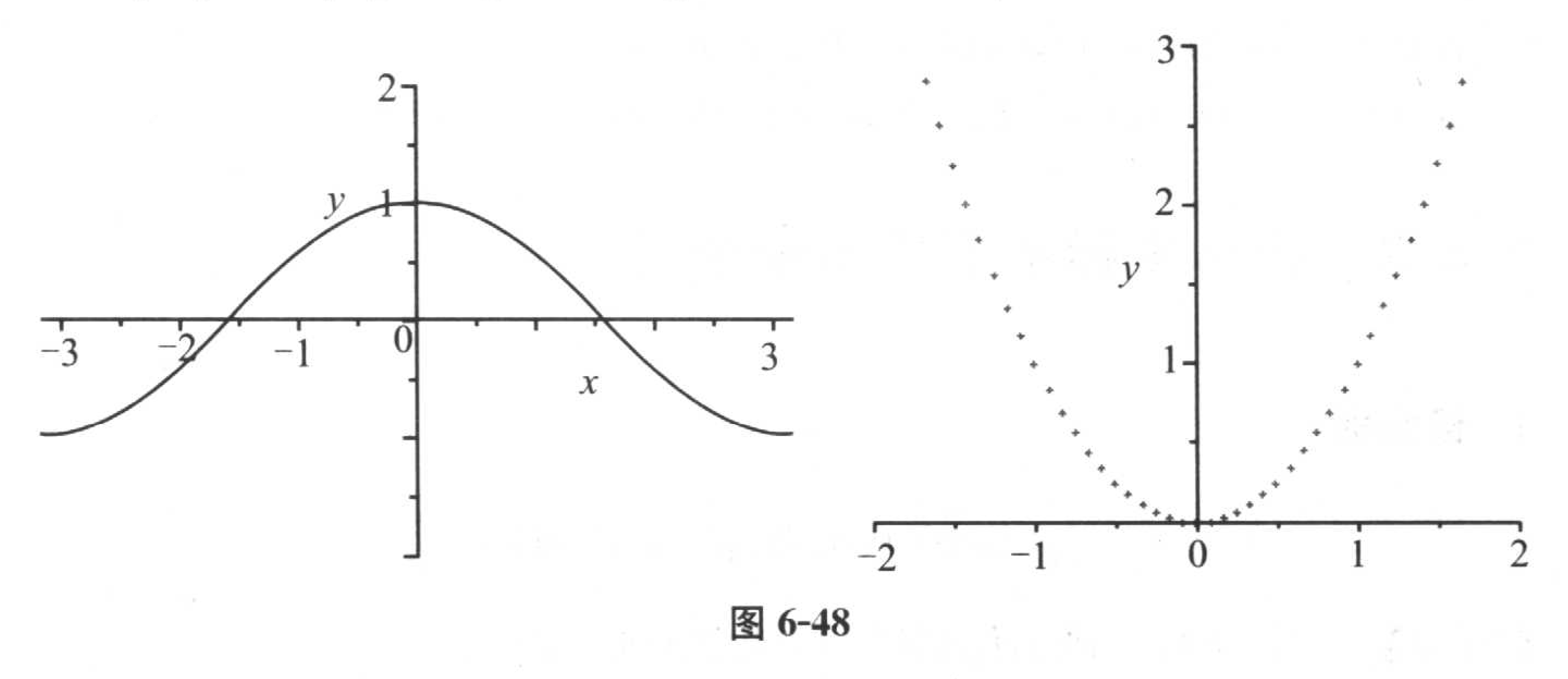


图 6-48

6.3.2 空间曲线

spacecurve(参数曲线,范围,选项)

【例 33】 画一条空间曲线(图 6-49)。

```
> with(plots):
> spacecurve([cos(t),sin(t),t],t=0..4*Pi);
```

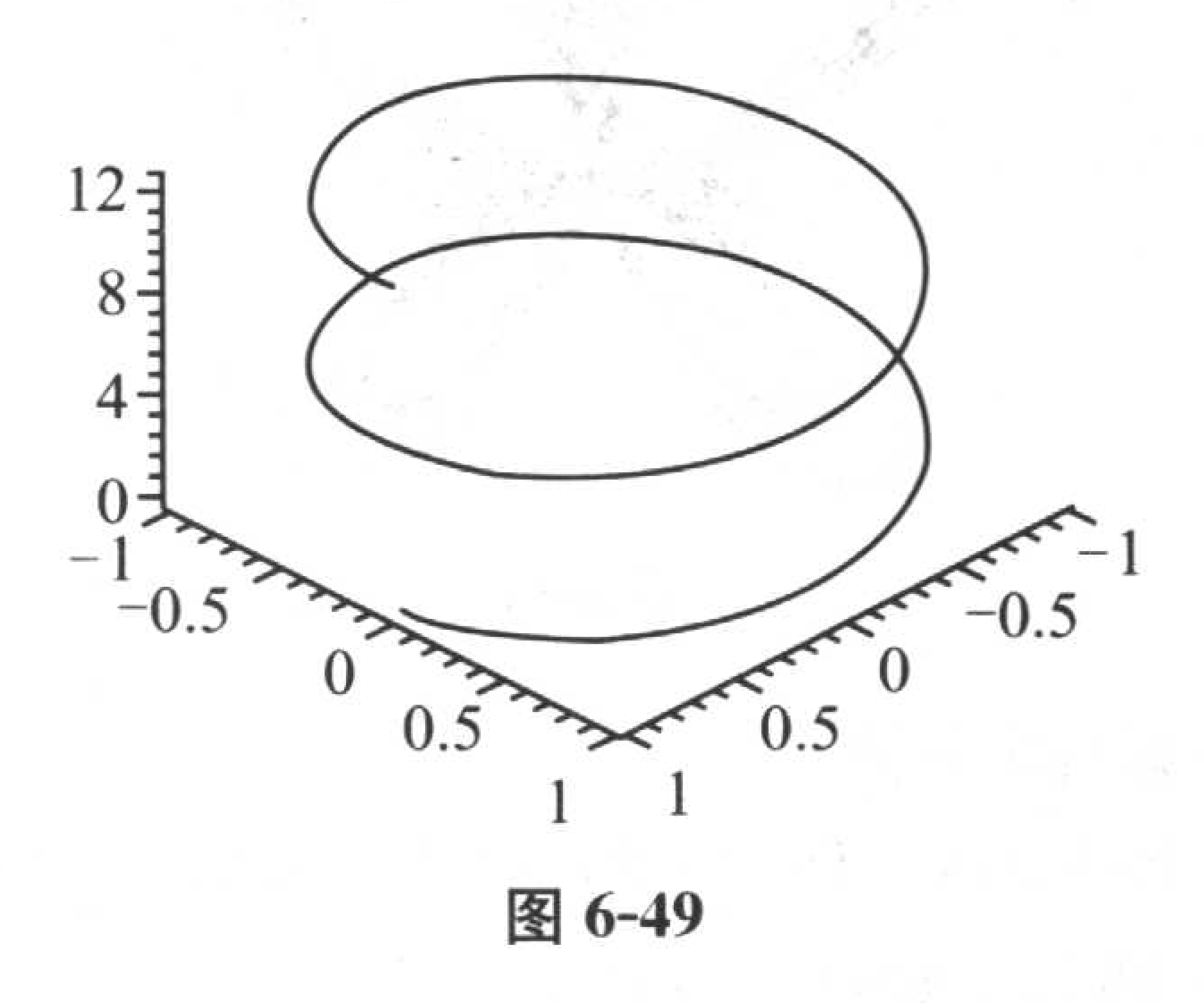


图 6-49

【例 34】 画一组空间曲线(图 6-50)。

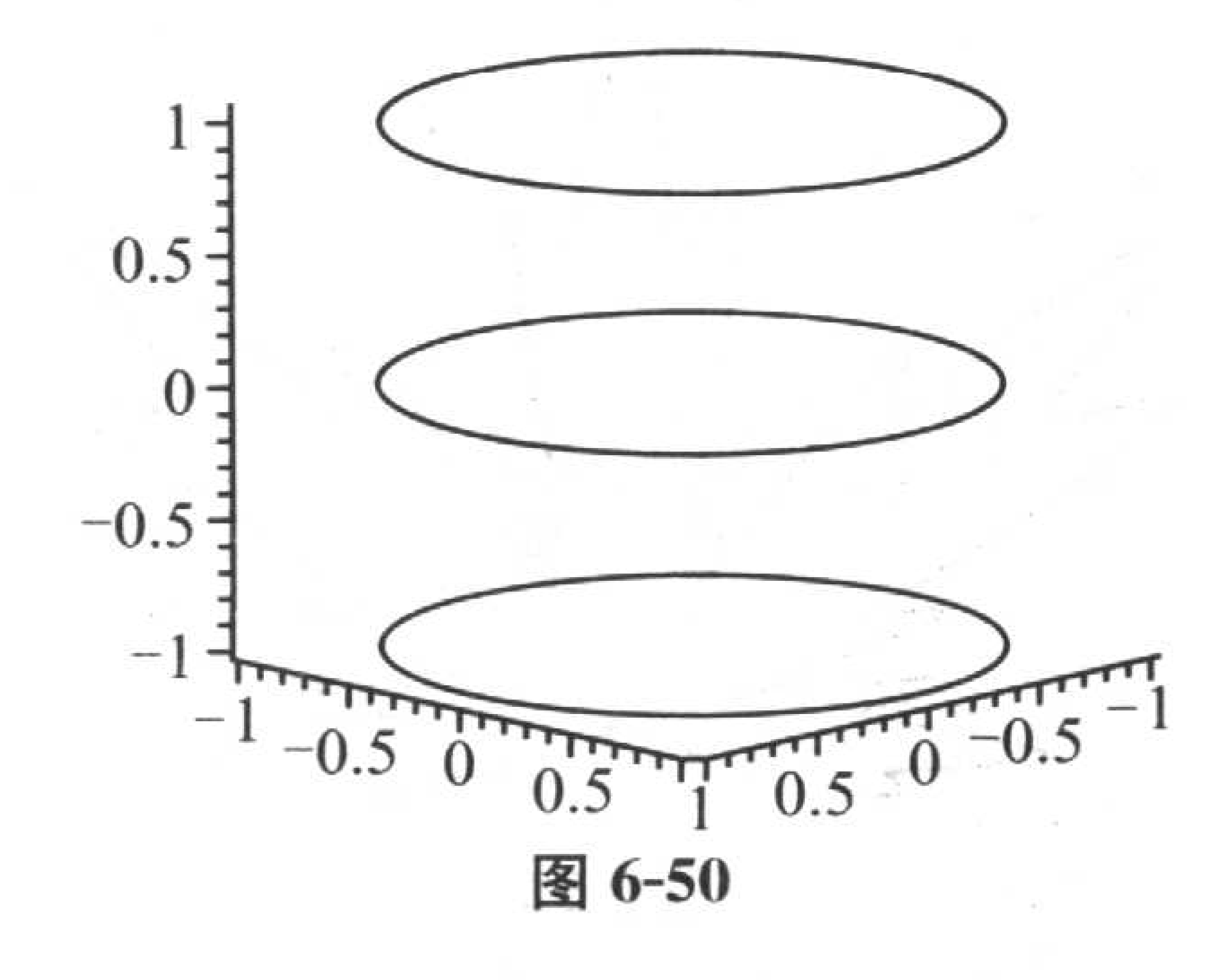


图 6-50

```
> spacecurve({[sin(t),cos(t),-1,t=0..2*Pi,color=red],
  [cos(2*t),sin(2*t),0,t=0..Pi,color=blue],
  [cos(t),sin(t),1,t=-Pi..Pi,color=black]});
```

6.3.3 特殊坐标系下作图命令

1. 极坐标

polarplot(函数,范围,选项)

【例 35】 在极坐标下画出像蝴蝶的函数图(图 6-51)。

```
> with(plots):
> polarplot(exp(cos(t-Pi/2))-2*cos(4*(t-Pi/2))+sin((t-Pi/2)/12)^5,
  t=0..36*Pi,axes=none);
```



图 6-51

【例 36】 画枫叶图案(图 6-52)。

```
> polarplot([100/(100+(t-Pi/2)^8)*(2-sin(7*t)-cos(30*t)/2), t,
  t=-Pi/2..3/2*Pi], axes=none);
```



图 6-52

2. 柱坐标

cylinderplot(函数,范围,选项)

【例 37】 简单的柱坐标函数图像(图 6-53,6-54)。

```
> cylinderplot(t,t=0..3*Pi,z=0..10,style=patchnogrid);
```

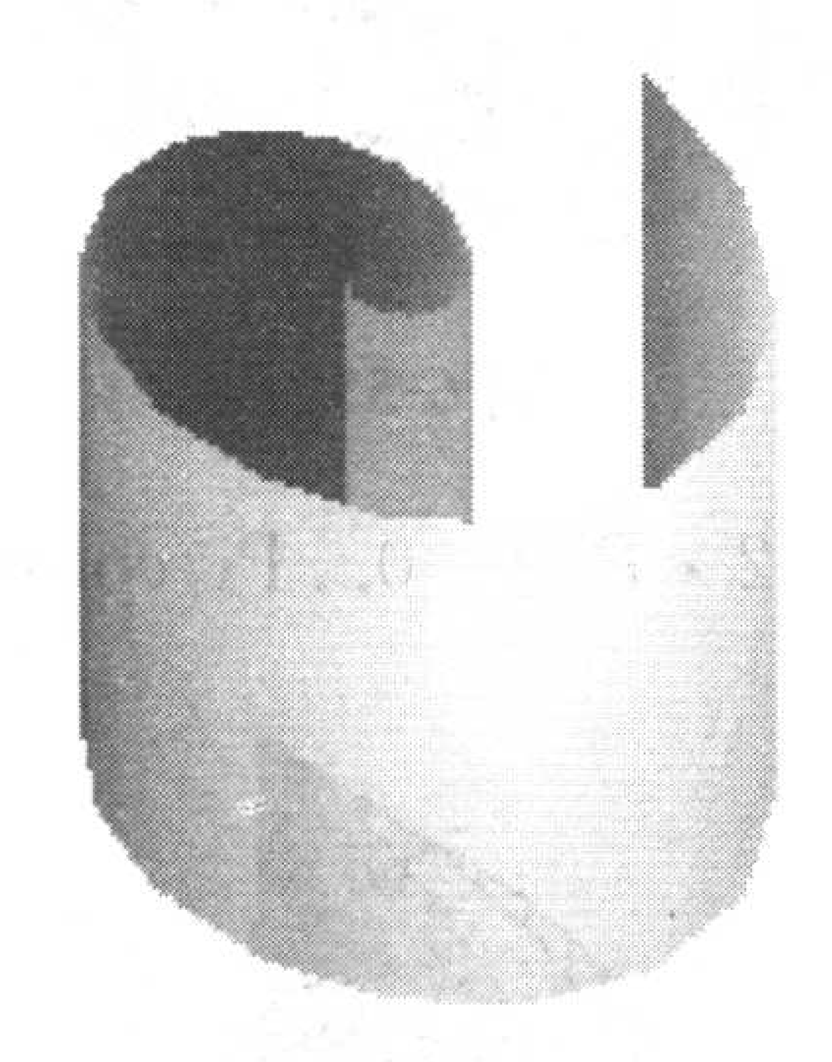


图 6-53

```
> cylinderplot(t+z,t=-Pi..Pi,z=0..5);
```

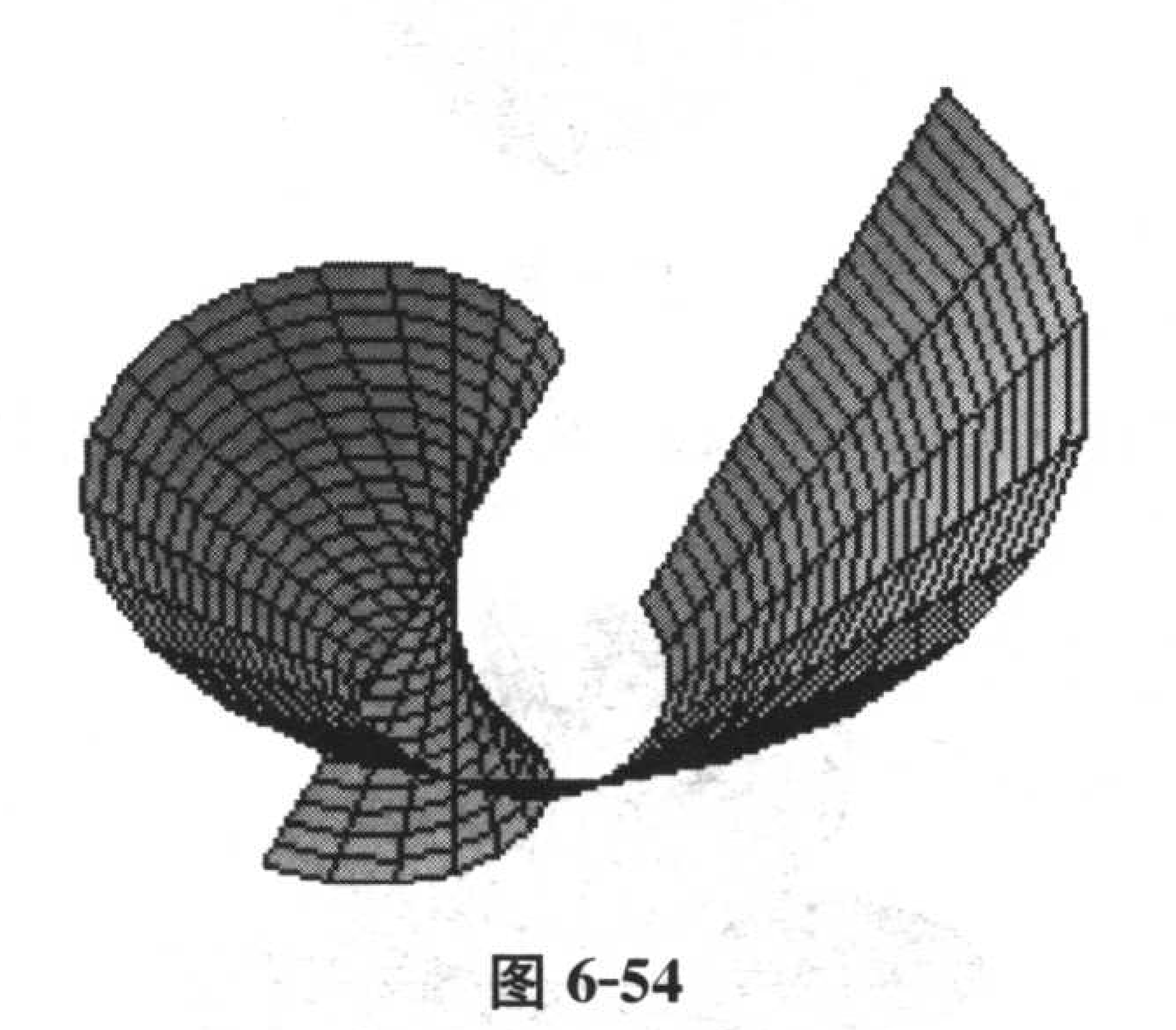


图 6-54

3. 球坐标

sphereplot(函数,范围,选项)

【例 38】 球坐标函数图(图 6-55～6-57)。

```
> sphereplot(cos(t),z=0..2*Pi,t=0..Pi,color=z);
```

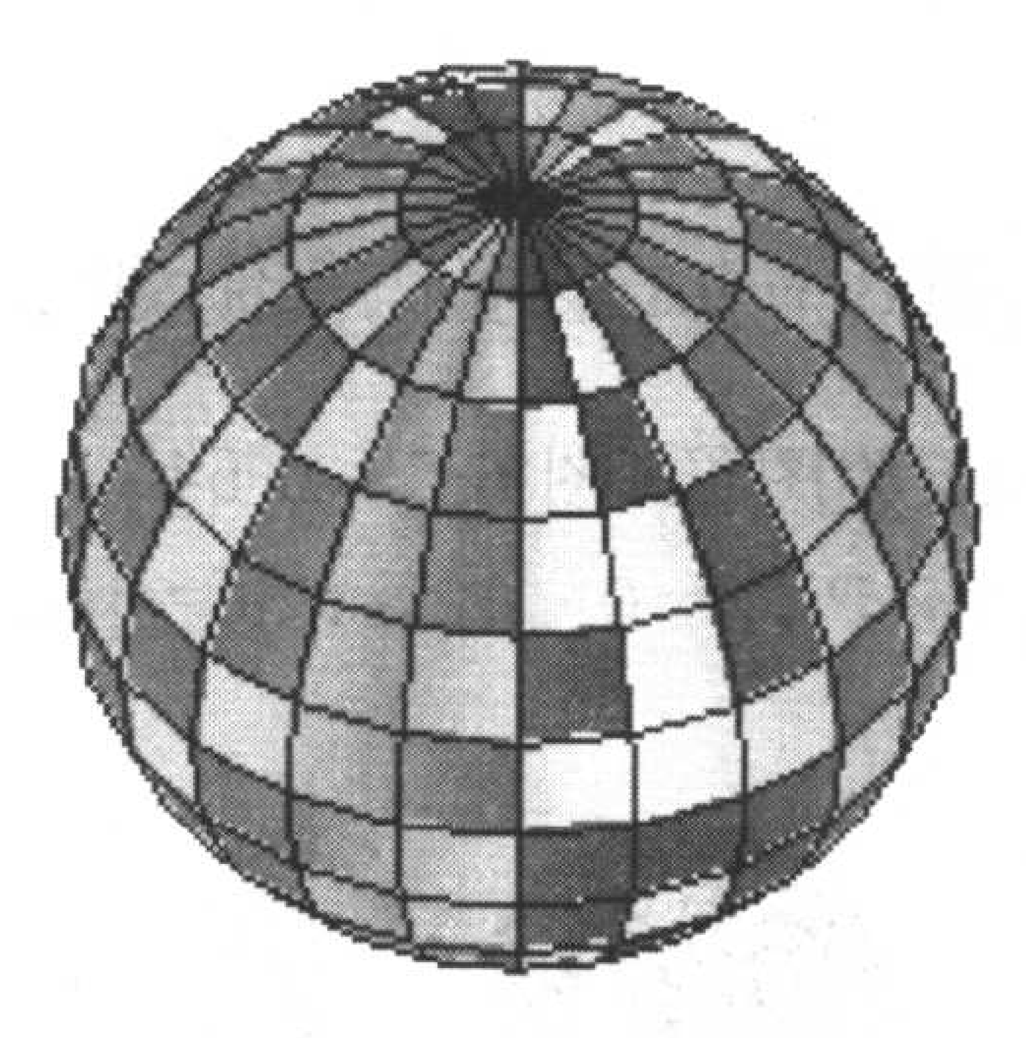


图 6-55

```
> sphereplot(cos(z),z=0..2*Pi,t=0..Pi,color=t);
```

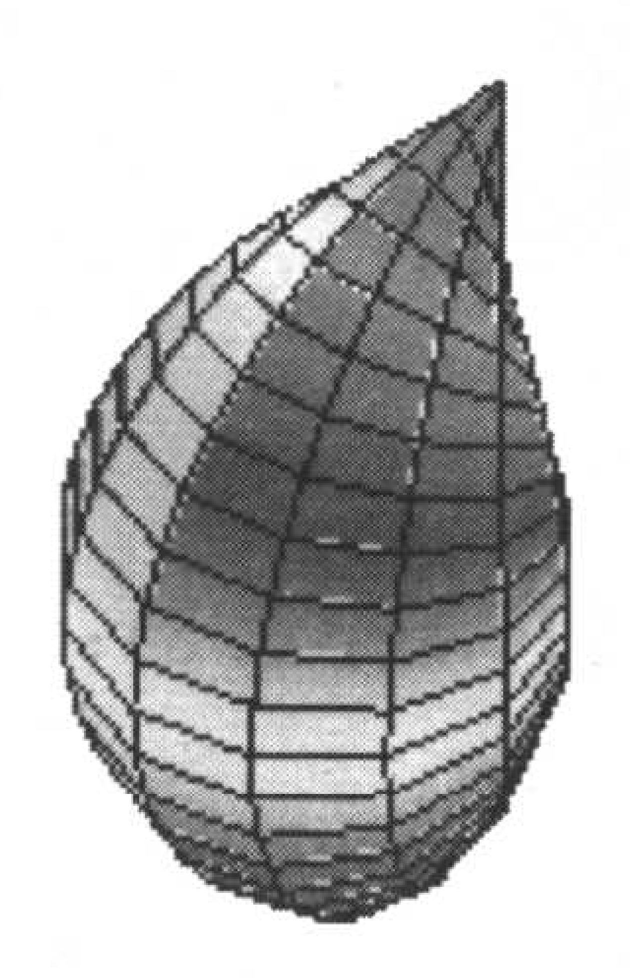


图 6-56

```
> sphereplot(cos(t^5)*sin(z/5),z=0..2*Pi,t=0..Pi);
```

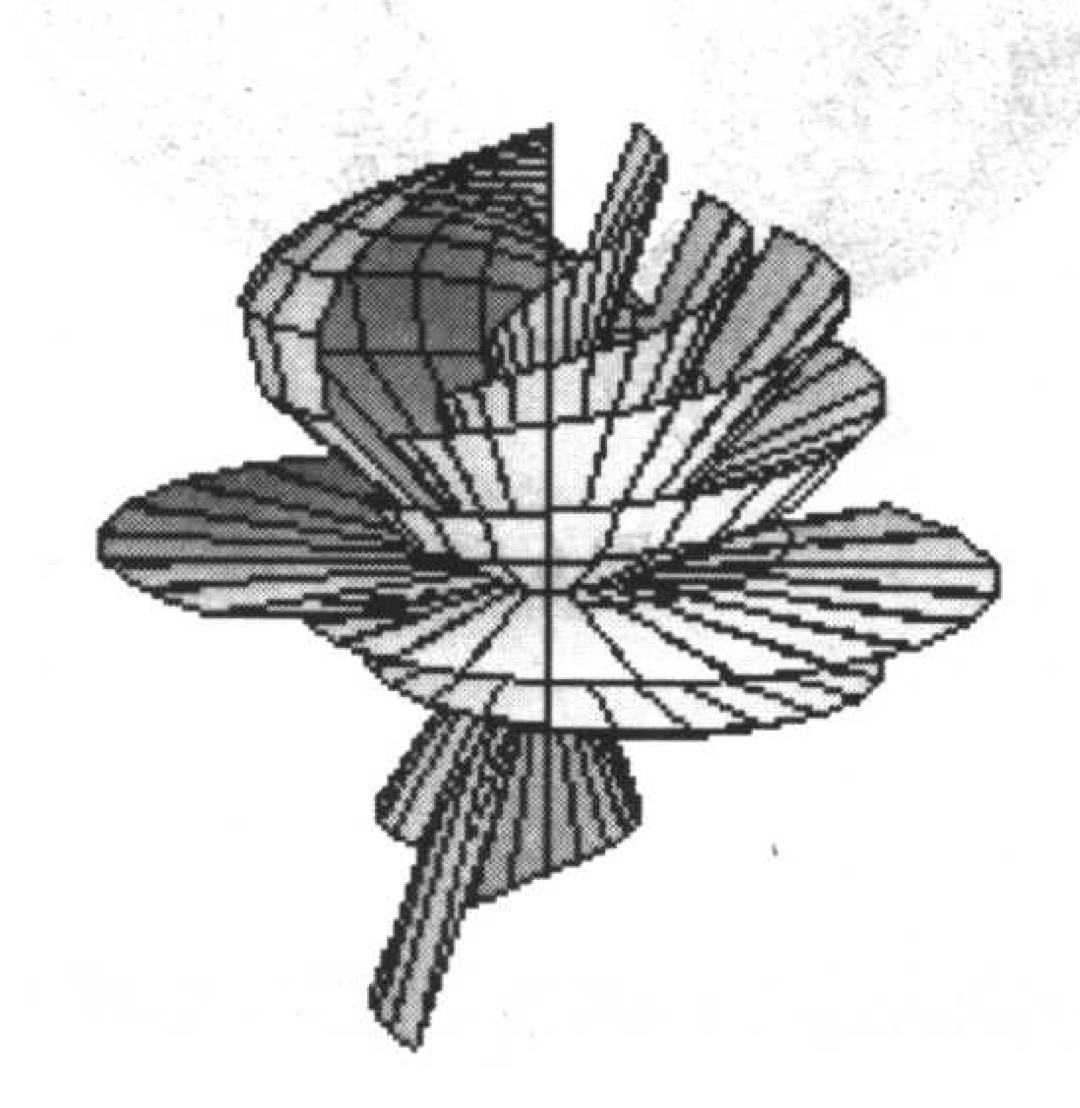


图 6-57

6.3.4　等值线图、密度图、向量场图和梯度图

1. 等值线图

contourplot(函数,范围,选项)
contourplot3d(函数,范围,选项)

【例 39】　函数 $f(x,y)=-\dfrac{3x}{x^2+y^2+1}$ 的等值线图(图 6-58～6-60)。

```
> with(plots):
> contourplot(-3*x/(x^2+y^2+1),x=-4..4,y=-4..4,filled=true);
  contourplot(-3*x/(x^2+y^2+1),x=-4..4,y=-4..4);
```

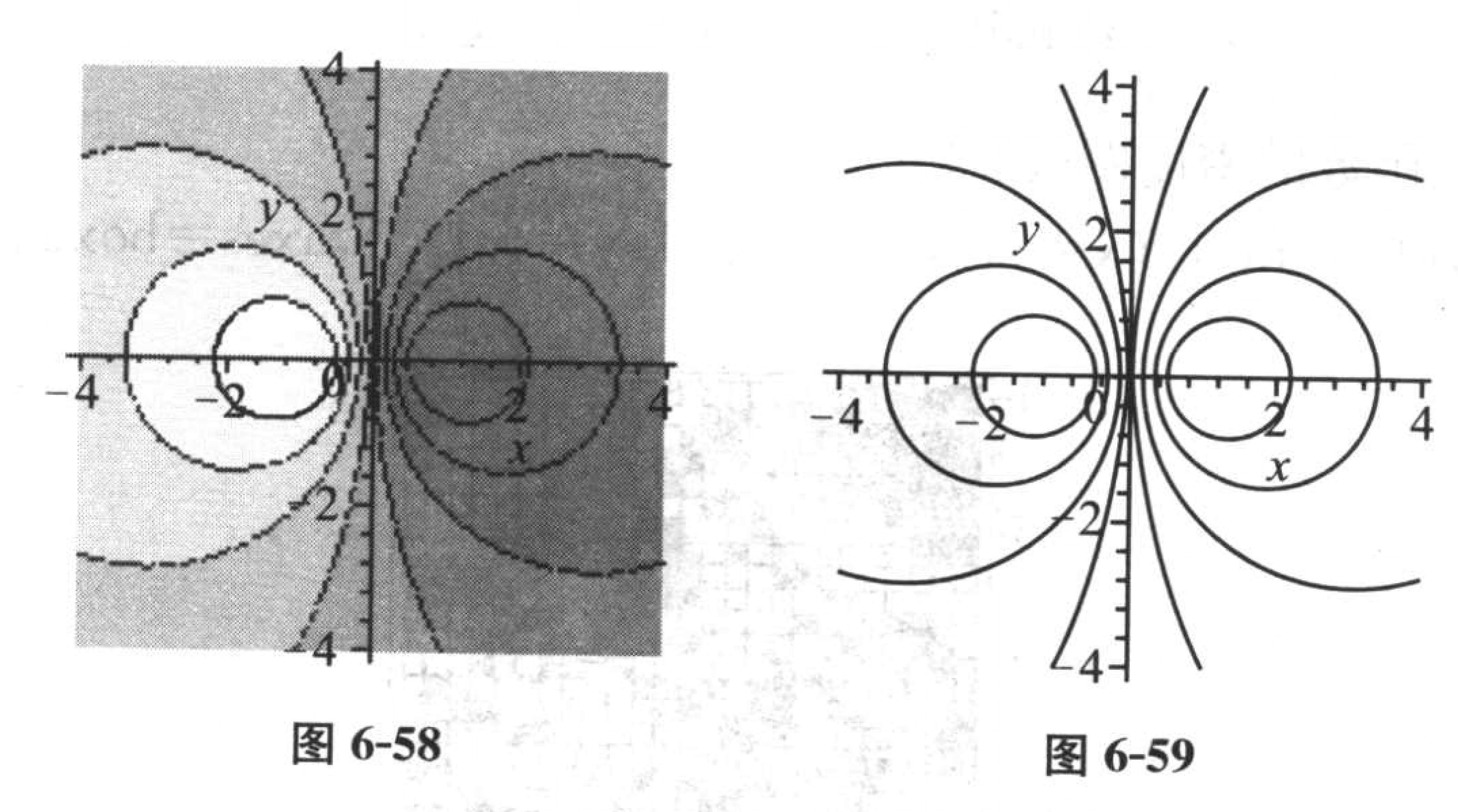


图 6-58　　图 6-59

```
> contourplot3d(-3*x/(x^2+y^2+1),x=-4..4,y=-4..4,filled=true);
```

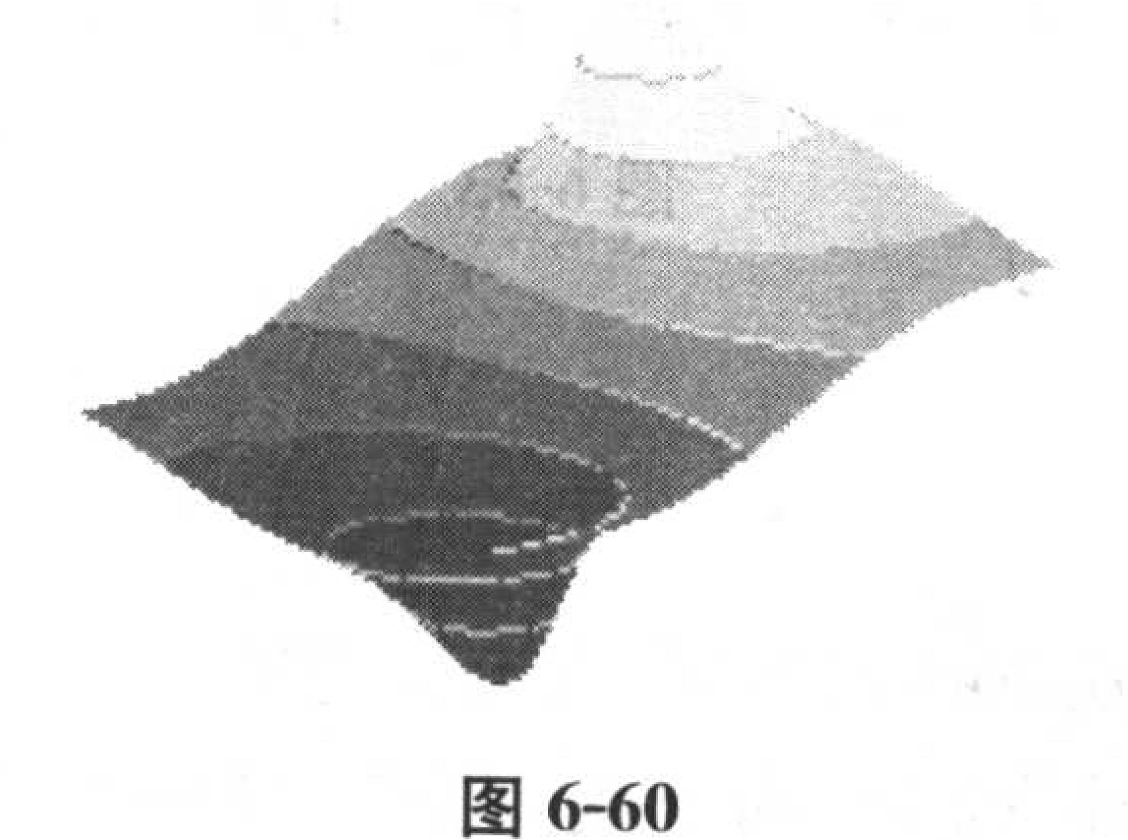


图 6-60

【例 40】　函数 $f(x)=\sin(\cos(x^2+y^2))$ 的等值线图(图 6-61,6-62)。

```
> contourplot(sin(cos(x^2+y^2)),x=-10..10,y=-10..10,filled=true);
  contourplot3d(sin(cos(x^2+y^2)),x=-10..10,y=-10..10,filled=true);
```

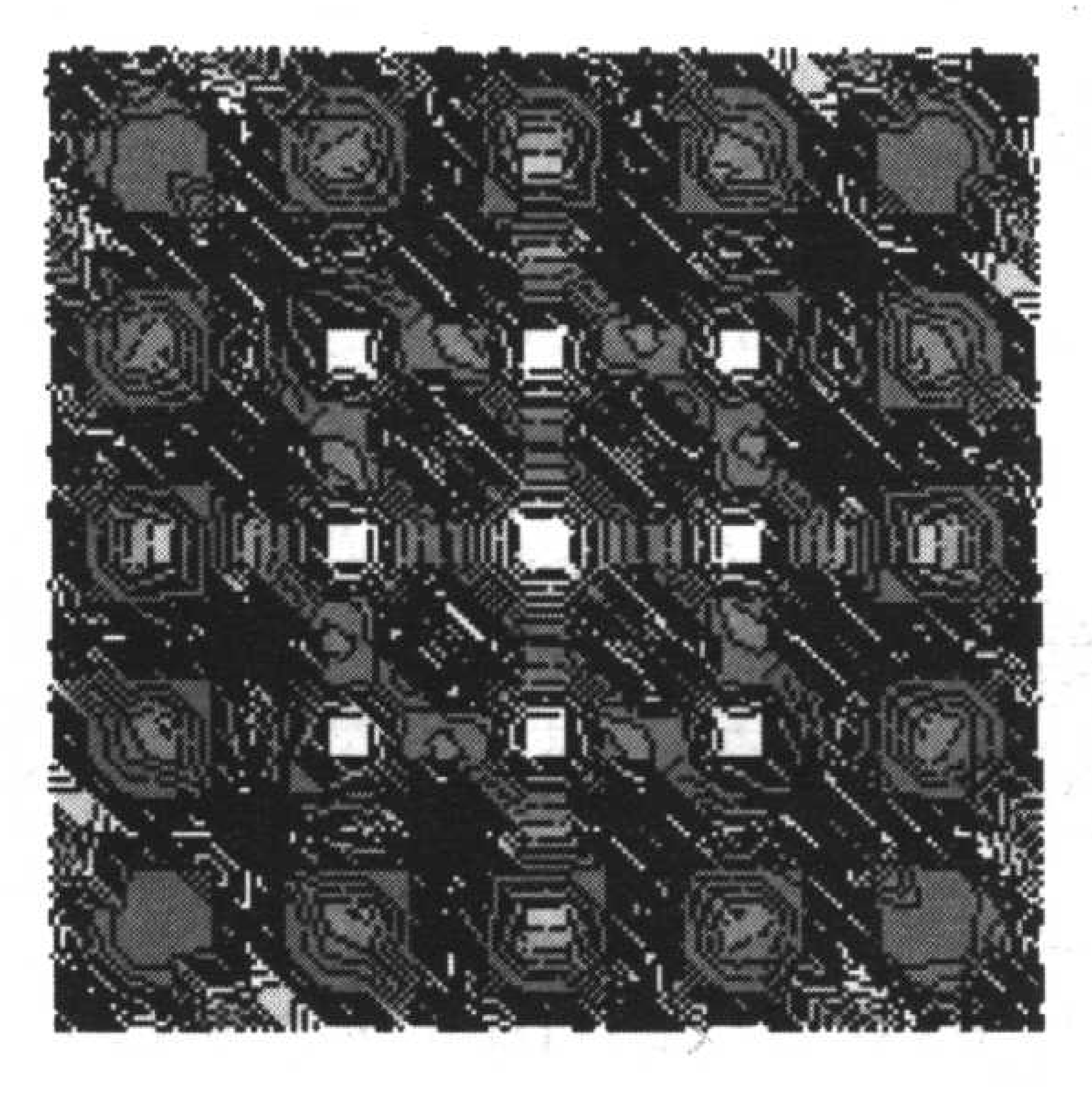


图 6-61

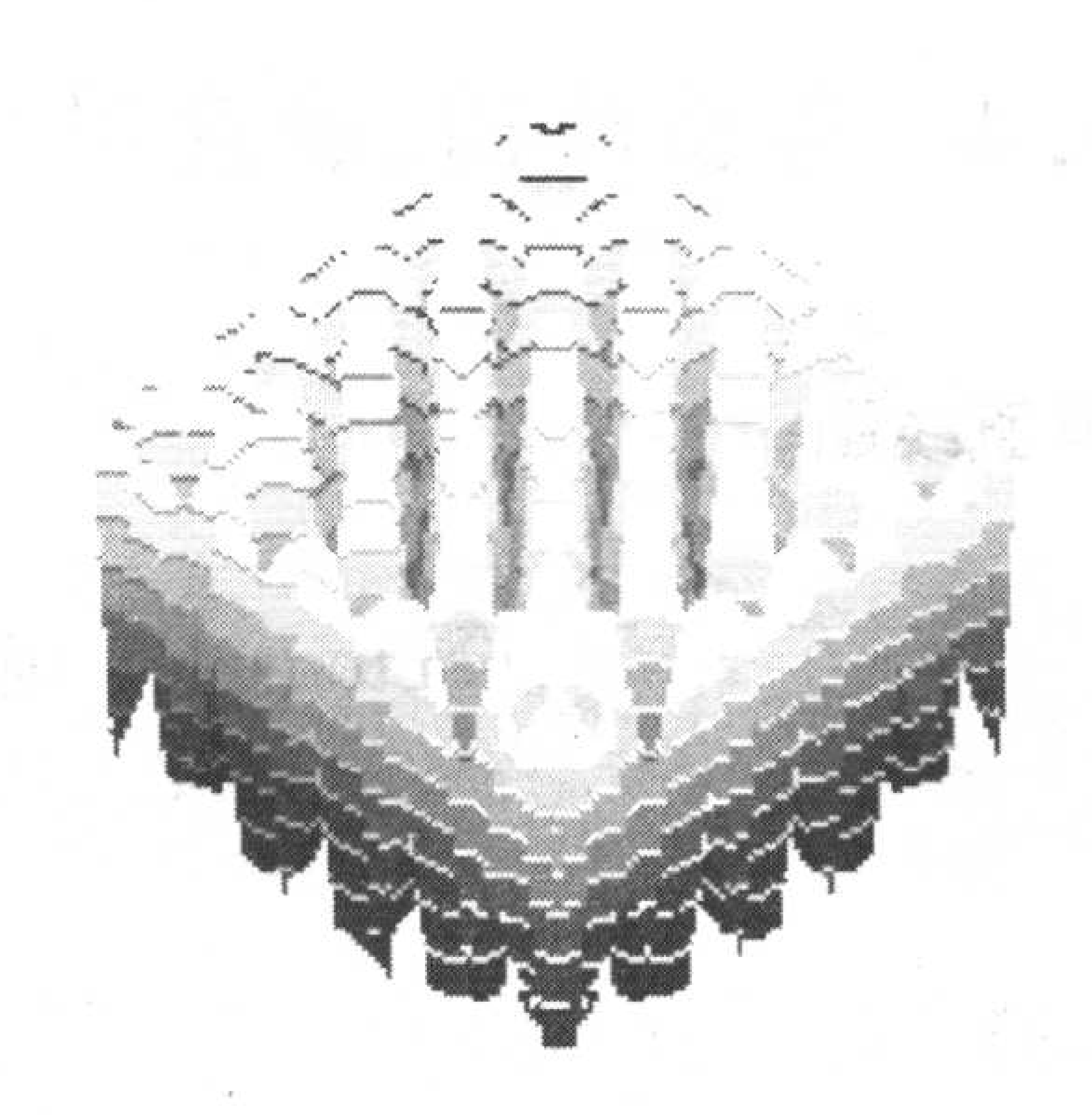


图 6-62

2. 密度图

densityplot(函数,范围,选项)

【例 41】 函数的密度图(图 6-63)。

> densityplot(sin(1/(x * y)),x=-1..1,y=-1..1,axes=boxed);

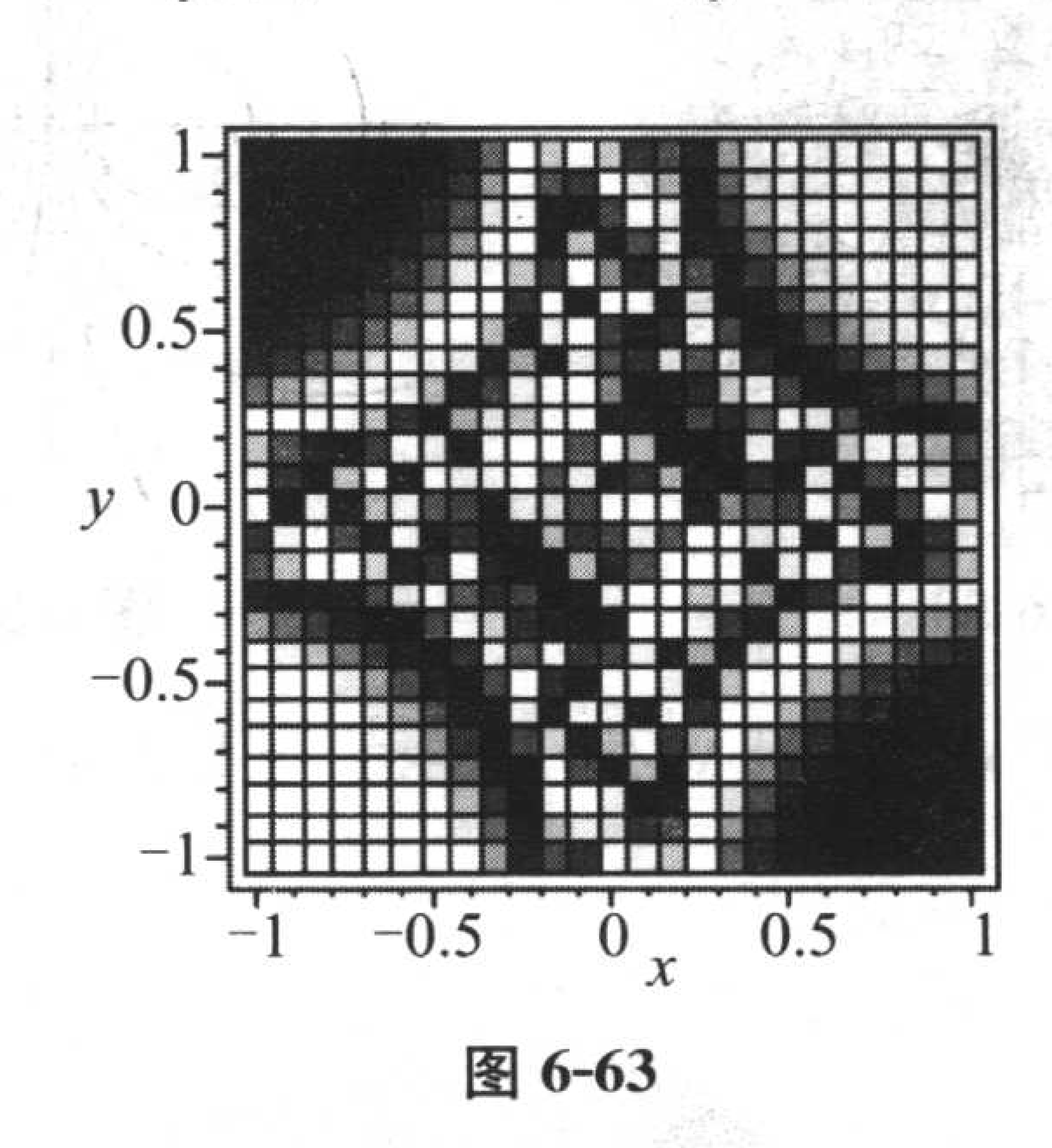


图 6-63

3. 向量场图

fieldplot(向量场,范围,选项)
fieldplot3d(向量场,范围,选项)

【例 42】 画二维向量场(图 6-64)。

> fieldplot([sin(2 * x * y),cos(2 * x-y)],x=-2..2,y=-2..2,axes=boxed);

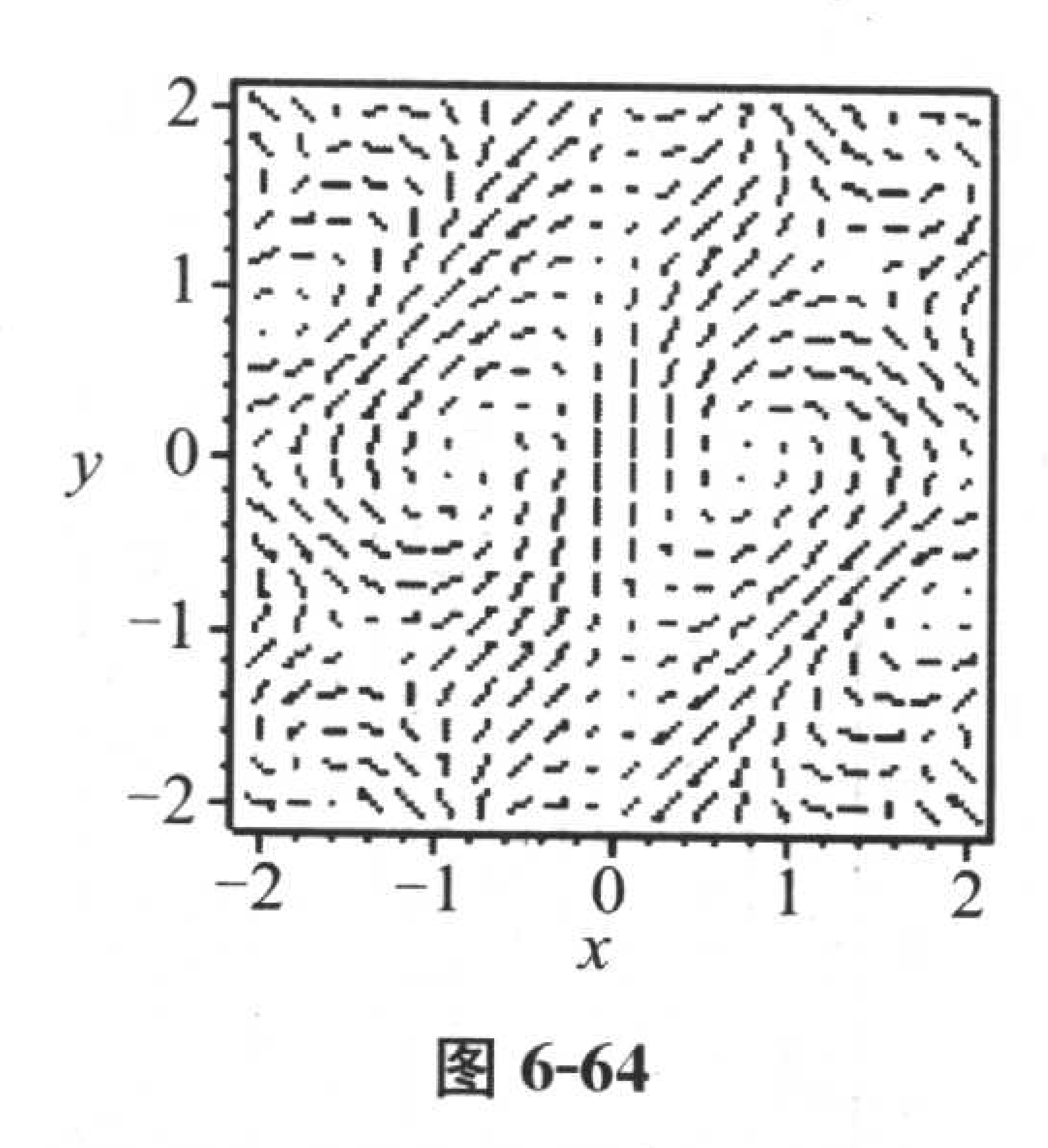


图 6-64

【例 43】 画三维向量场(图 6-65)。

> fieldplot3d([x^2,y*2,z+2],x=-1..1,y=-1..1,z=-1..1,axes=boxed);

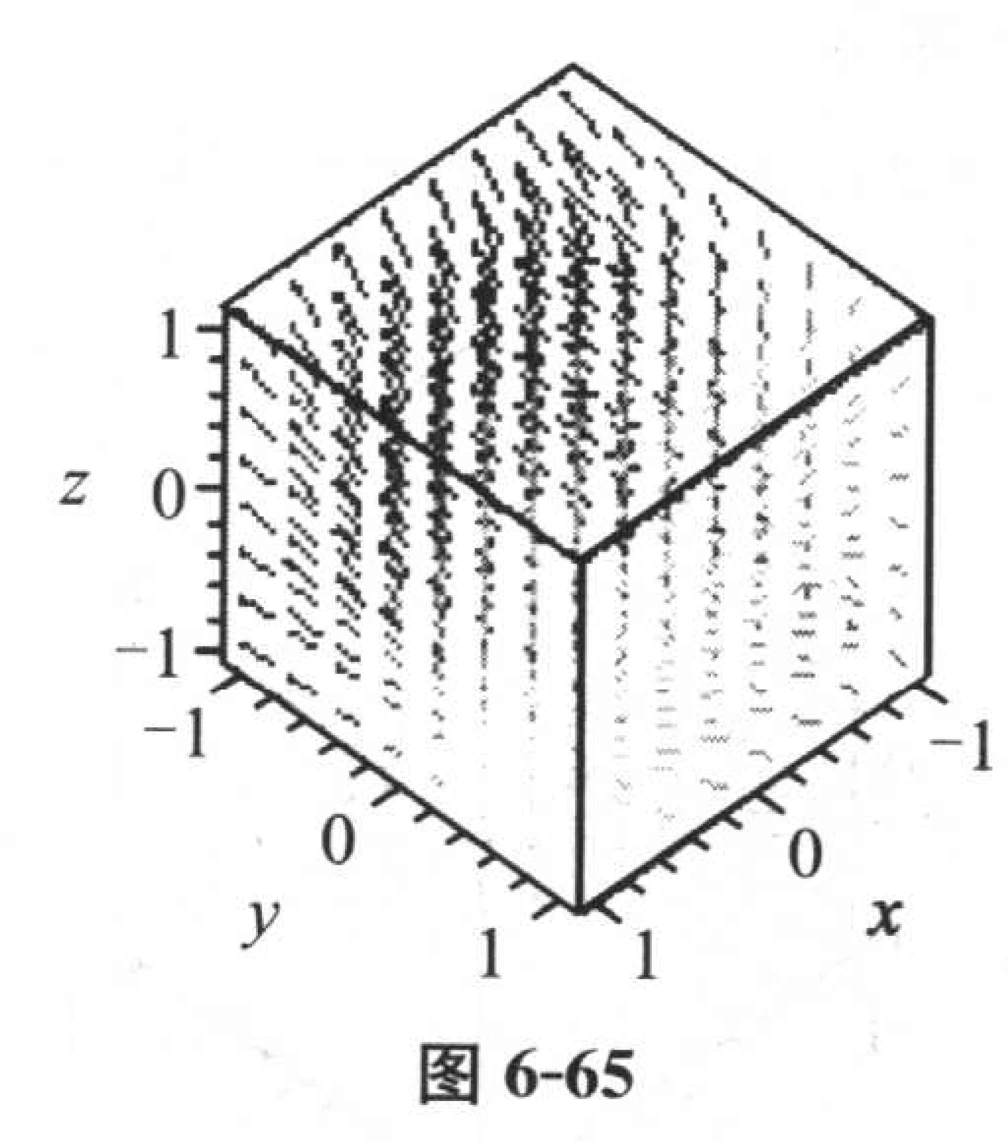


图 6-65

4. 梯度图

gradplot(函数,范围,选项)
gradplot3d(函数,范围,选项)

【例 44】 画二维梯度图(图 6-66,6-67)。

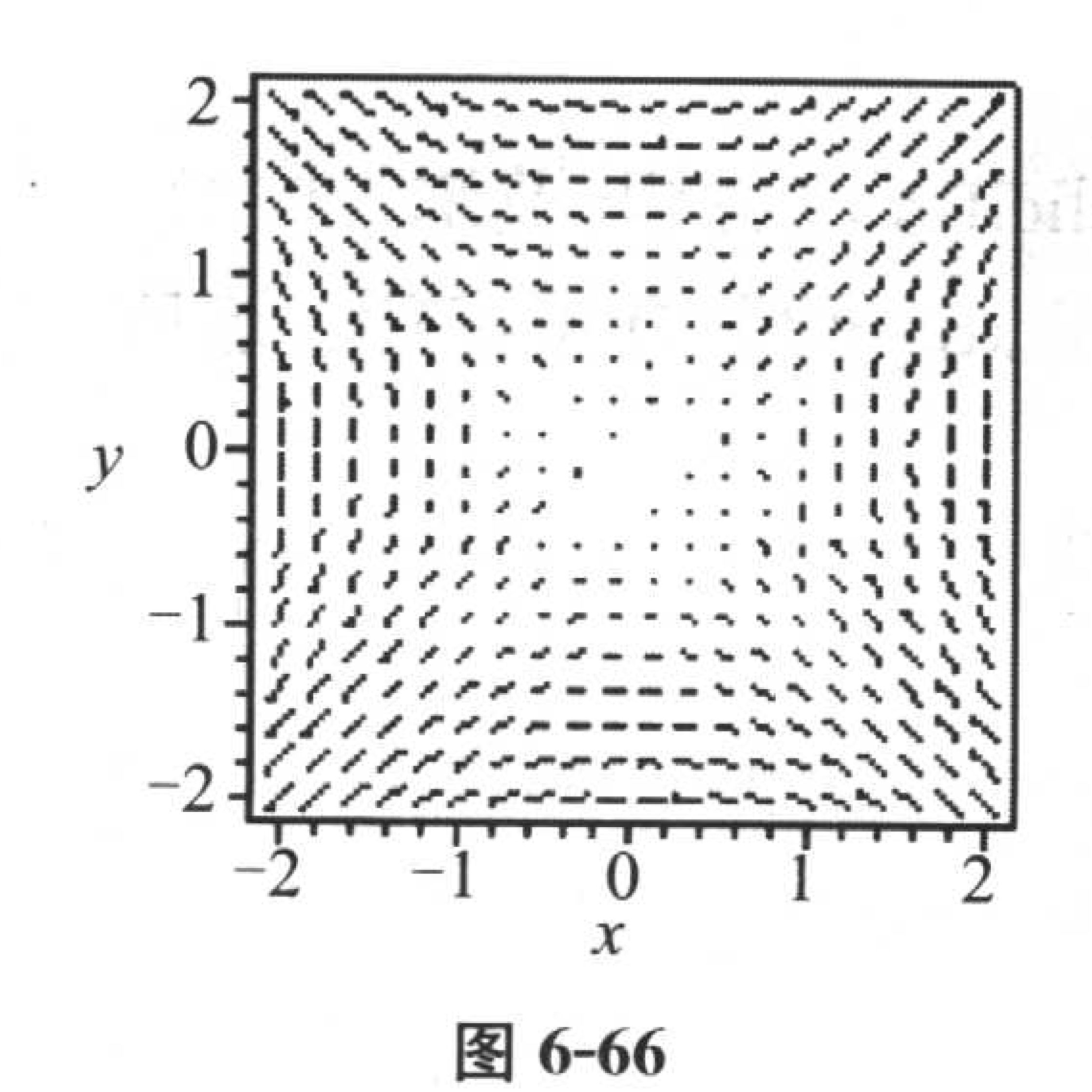


图 6-66

```
> gradplot(x * y,x=-2..2,y=-2..2,axes=boxed);
> gradplot(sin(3 * x) * cos(3 * y),x=-2..2,y=-2..2);
```

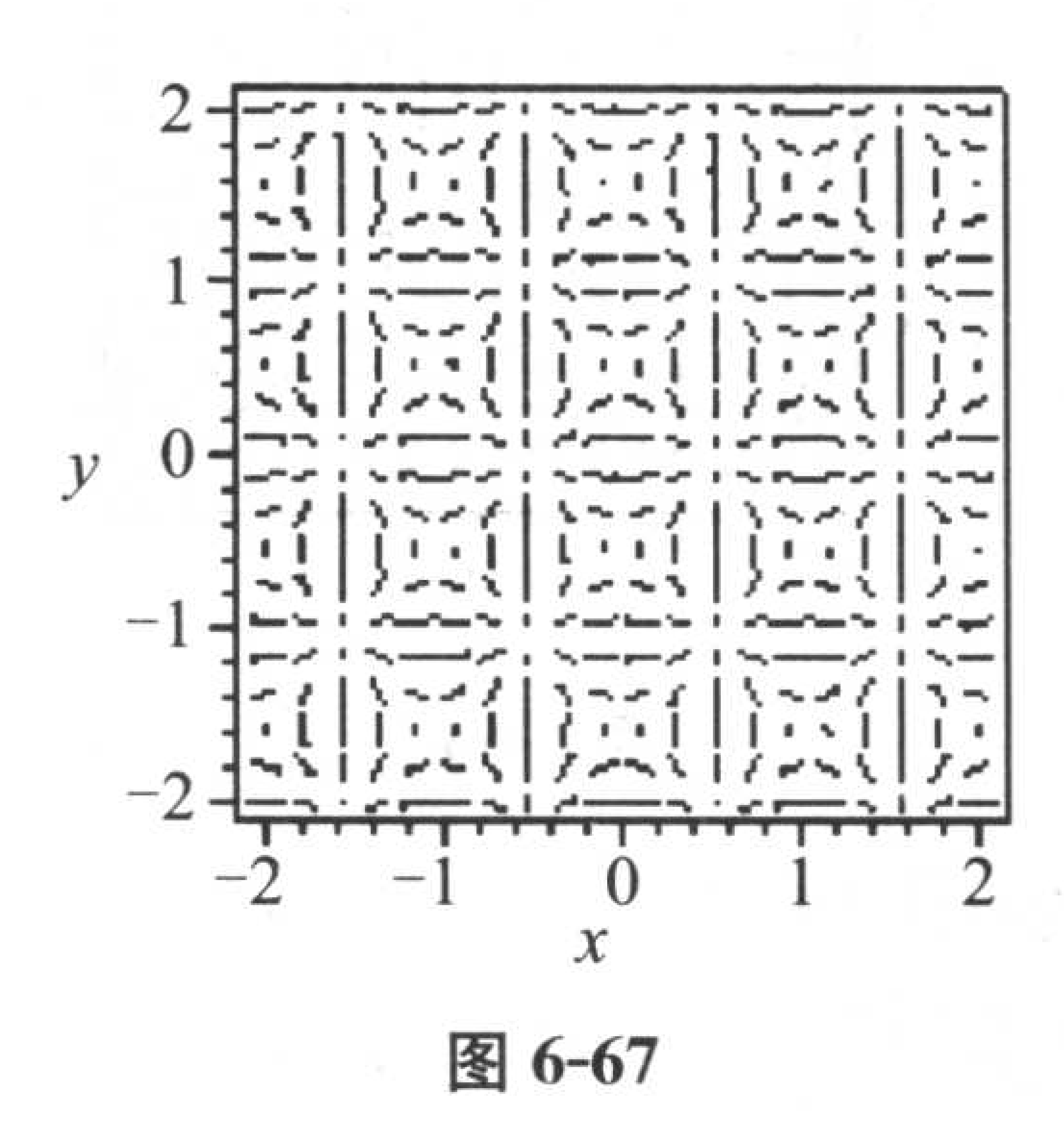


图 6-67

【例 45】 画三维梯度图(图 6-68)。

```
> gradplot3d(x^2+y^2+z^2,x=-1..1,y=-1..1,z=-1..1,axes=boxed);
```

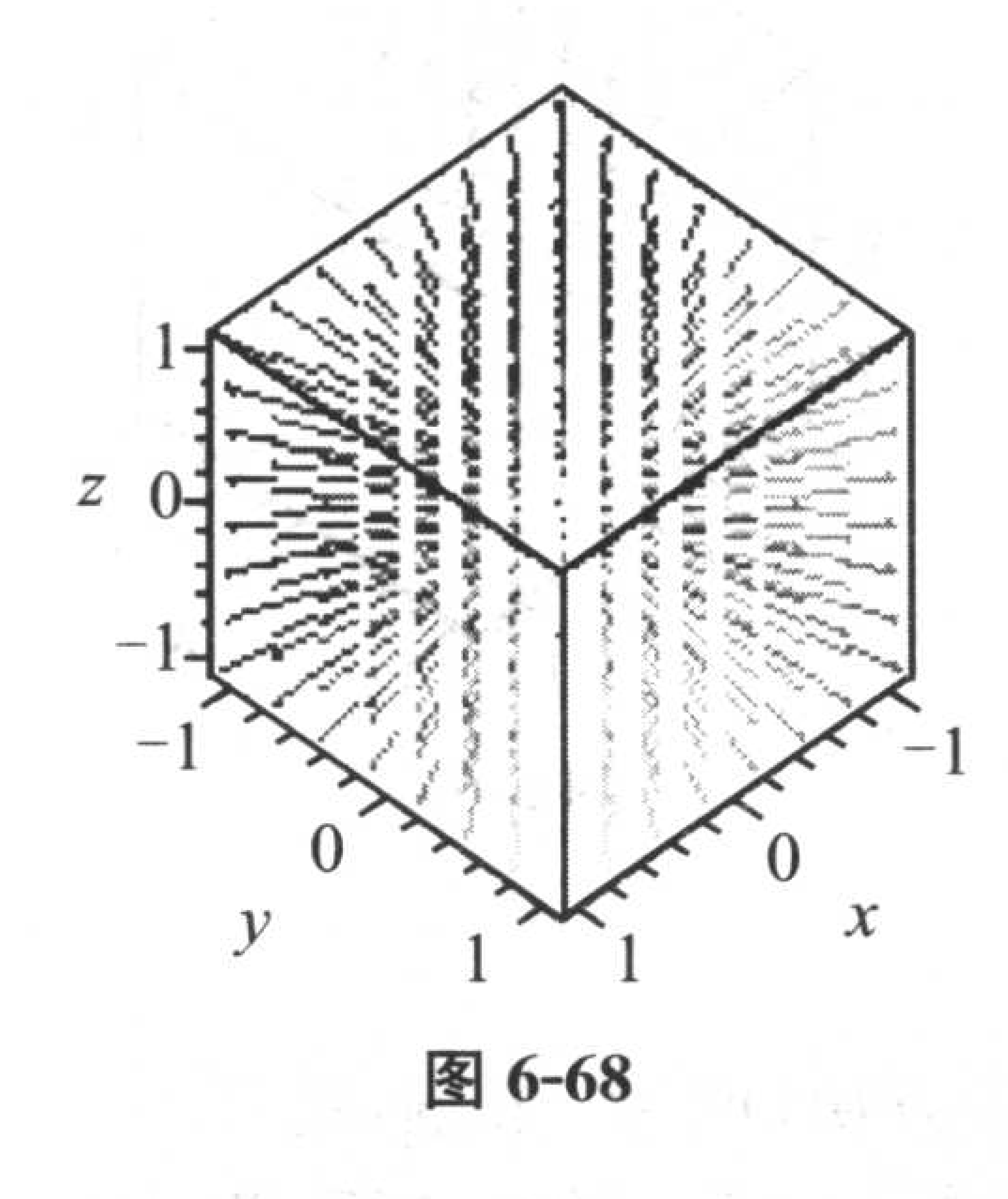


图 6-68

6.3.5 隐函数作图

> implicitplot(方程,范围,选项)
> implicitplot3d(方程,范围,选项)

【例 46】 画出二次曲线 $x^2 - y^2 = 1$ 和 $x^2 + y^2 = 1$(图 6-69)。

```
> with(plots):
> implicitplot([x^2-y^2=1,x^2+y^2=1],x=-Pi..Pi,y=-Pi..Pi,
  color=[blue,red], legend=["x^2-y^2=1","x^2+y^2=1"]);
```

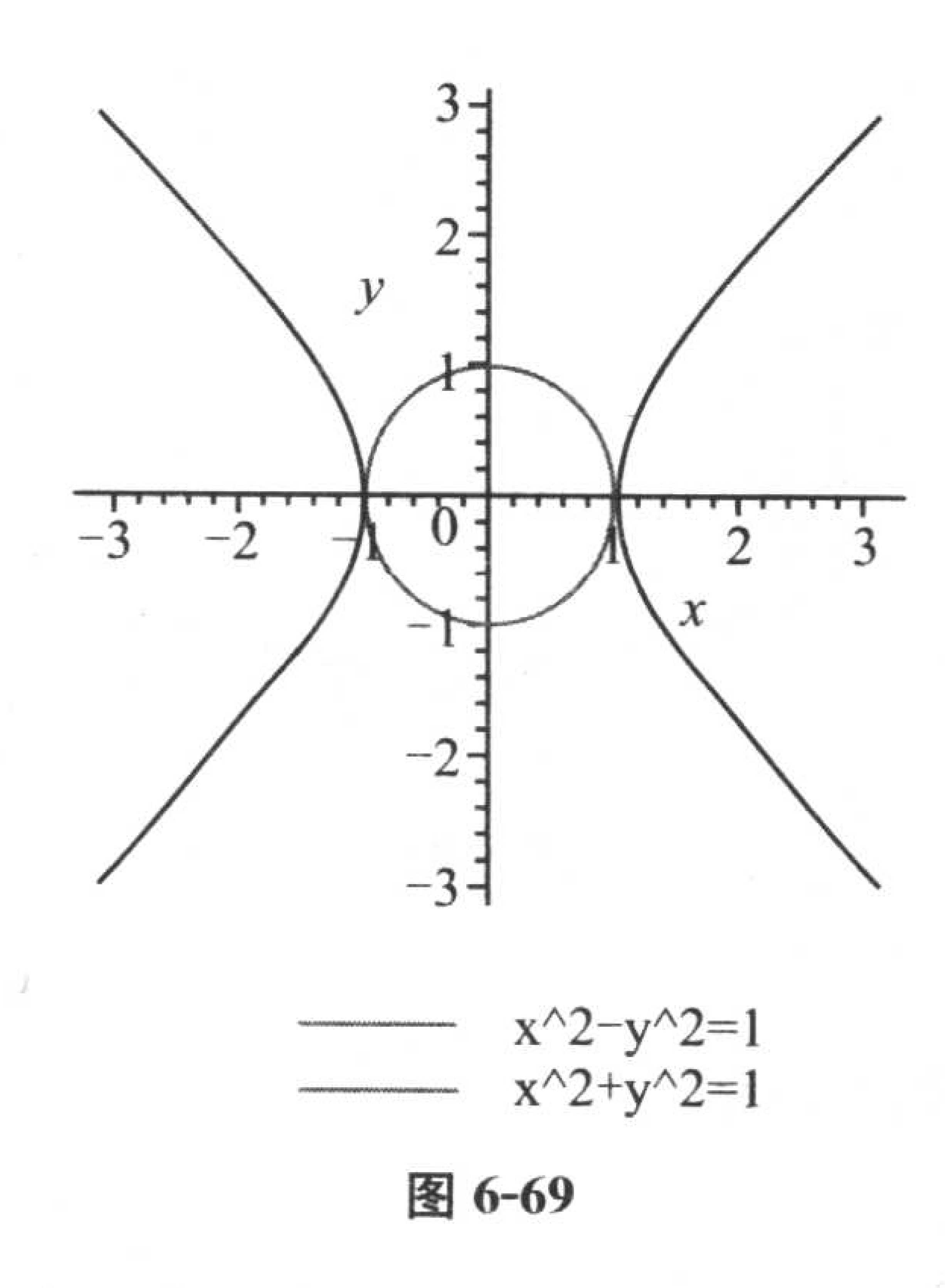


图 6-69

【例 47】　画出函数 $x^2+y^2=z^2$ 的图形(图 6-70)。

```
> implicitplot3d(x^2+y^2=z^2,x=-1..1,y=-1..1,z=-1..1);
```

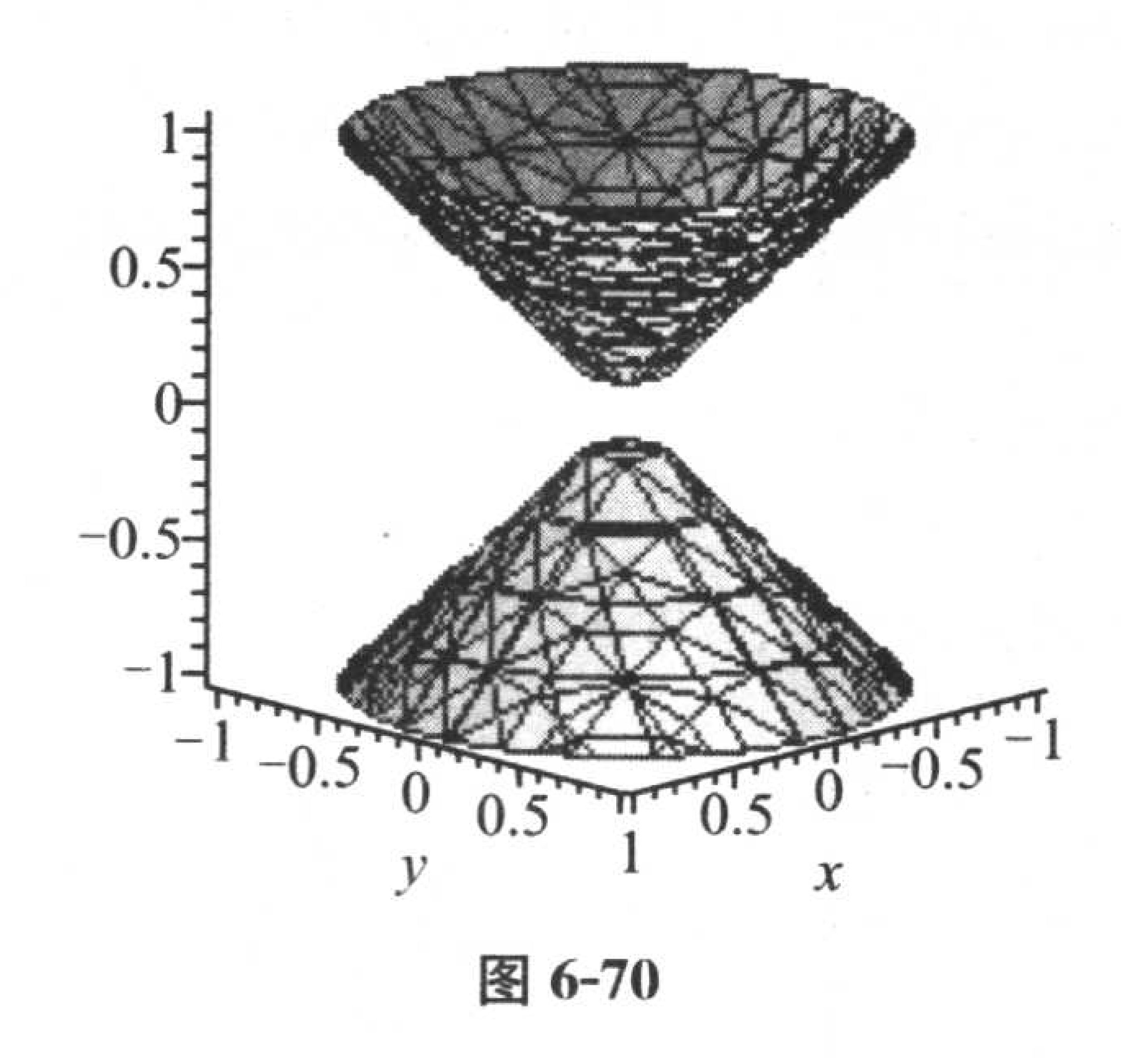


图 6-70

【例 48】　画出函数 $x^3+y^3+z^3+1=(x+y+z+1)^3$ 的图形(图 6-71)。

```
> implicitplot3d(x^3+y^3+z^3+1=(x+y+z+1)^3, x=-2..2, y=-2..2,
  z=-2..2, style=patchnogrid, numpoints=10000);
```

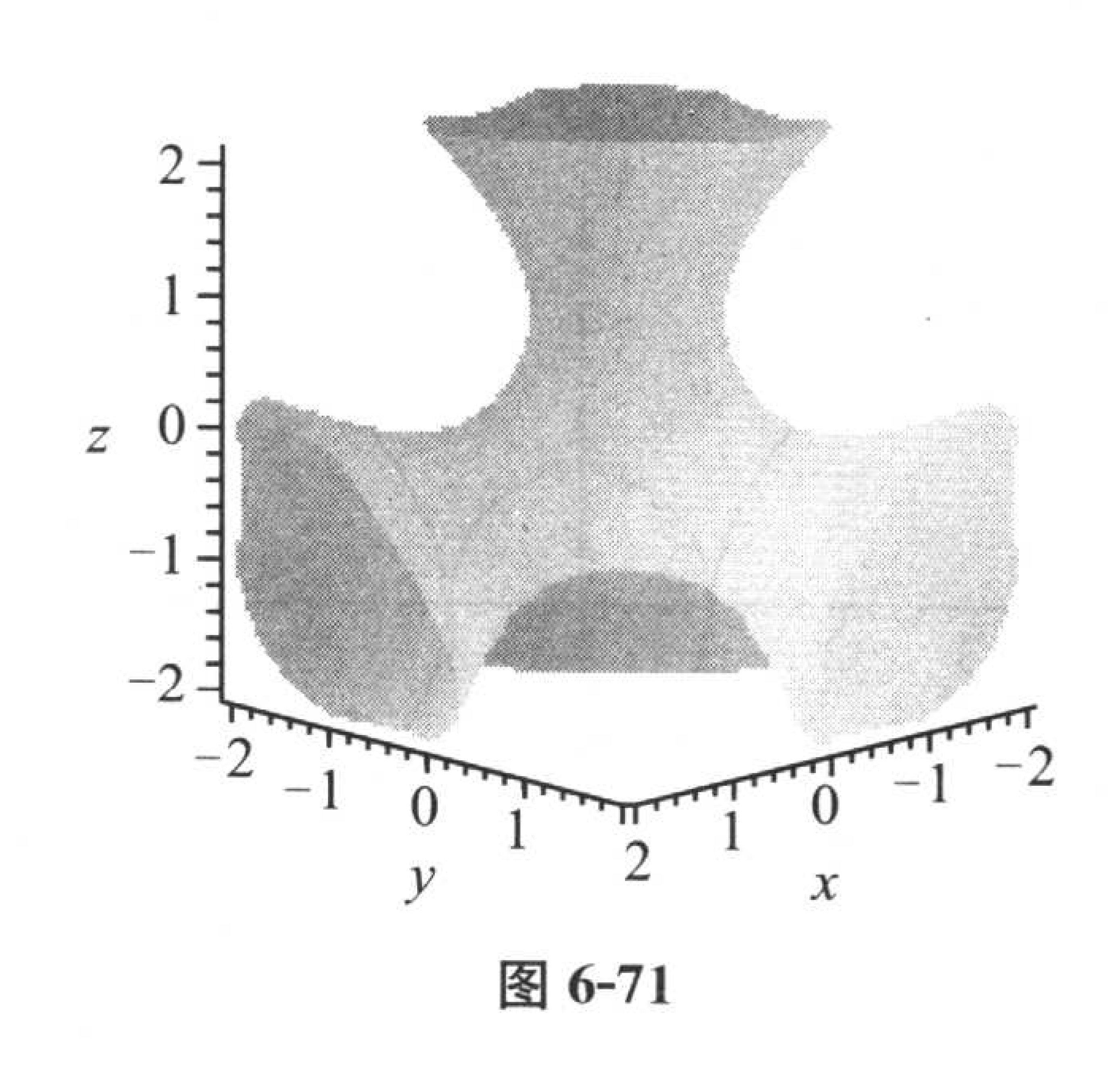


图 6-71

6.3.6 数据点列作图

```
listplot(数据点列,范围,选项)
listplot3d(数据点列,范围,选项)
listcontplot(数据点列,范围,选项)
listcontplot3d(数据点列,范围,选项)
listdensityplot(数据点列,范围,选项)
```

【例 49】 画折线图(图 6-72)。

```
> with(plots):
> listplot([1,3,2,5,4,9,6,10,7]);
```

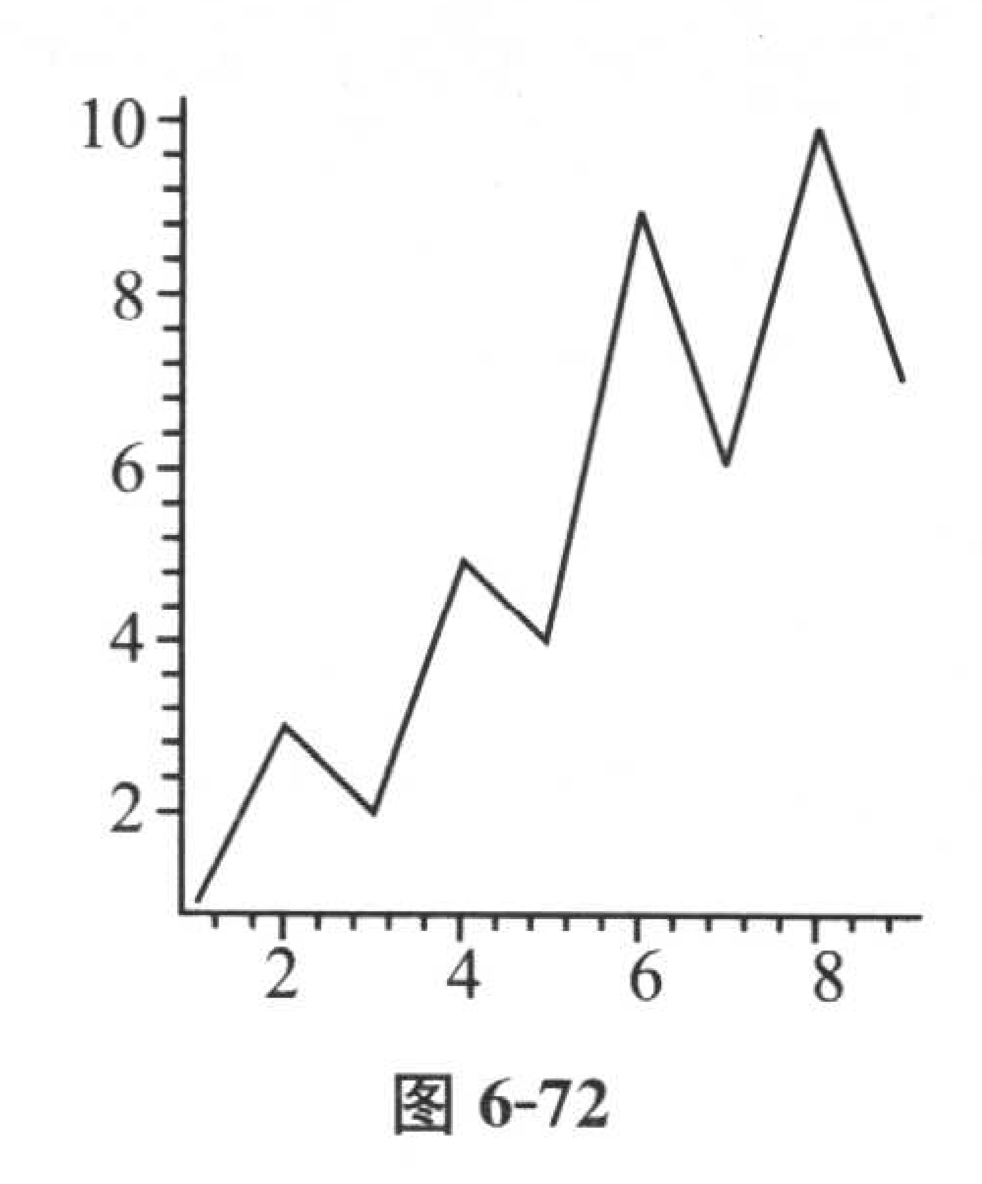


图 6-72

plot3d 命令按函数构造空间曲面,surfdata 命令用数据组构造空间曲面。

```
surfdata(数据点列,范围,选项)
```

【例 50】 画由两组数据生成的两张空间曲面(图 6-73)。

```
> d1 := seq([seq([i,j,cos((i+j)/5.)],i=-5..5)],j=-5..5):
  d2 := seq([seq([i,j,sin((i+j)/5.)],i=-5..5)],j=-5..5):
  surfdata([[d1],[d2]],axes=frame,labels=[x,y,z]);
```

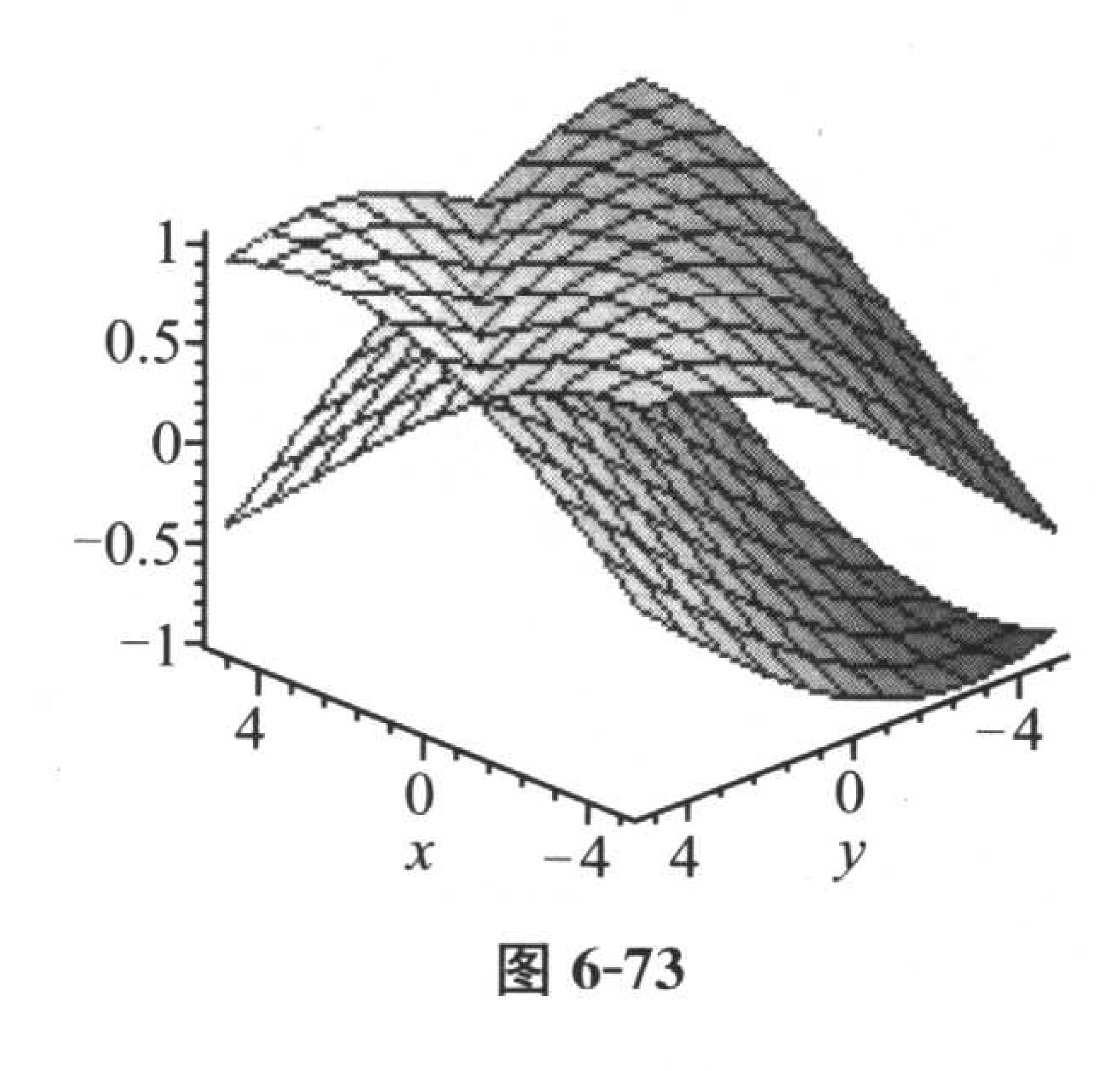


图 6-73

6.3.7 复函数作图

complexplot(函数,范围,选项)
complexplot3d(函数,范围,选项)
conformal(解析函数,范围,选项)
conformal3d(解析函数,范围,选项)

【例 51】 复函数 $\cos(x+i)$(图 6-74)、$\sec(z)$ 的图像(图 6-75)。

```
> with(plots):
> complexplot(cos(x+I),x=-Pi..Pi);
```

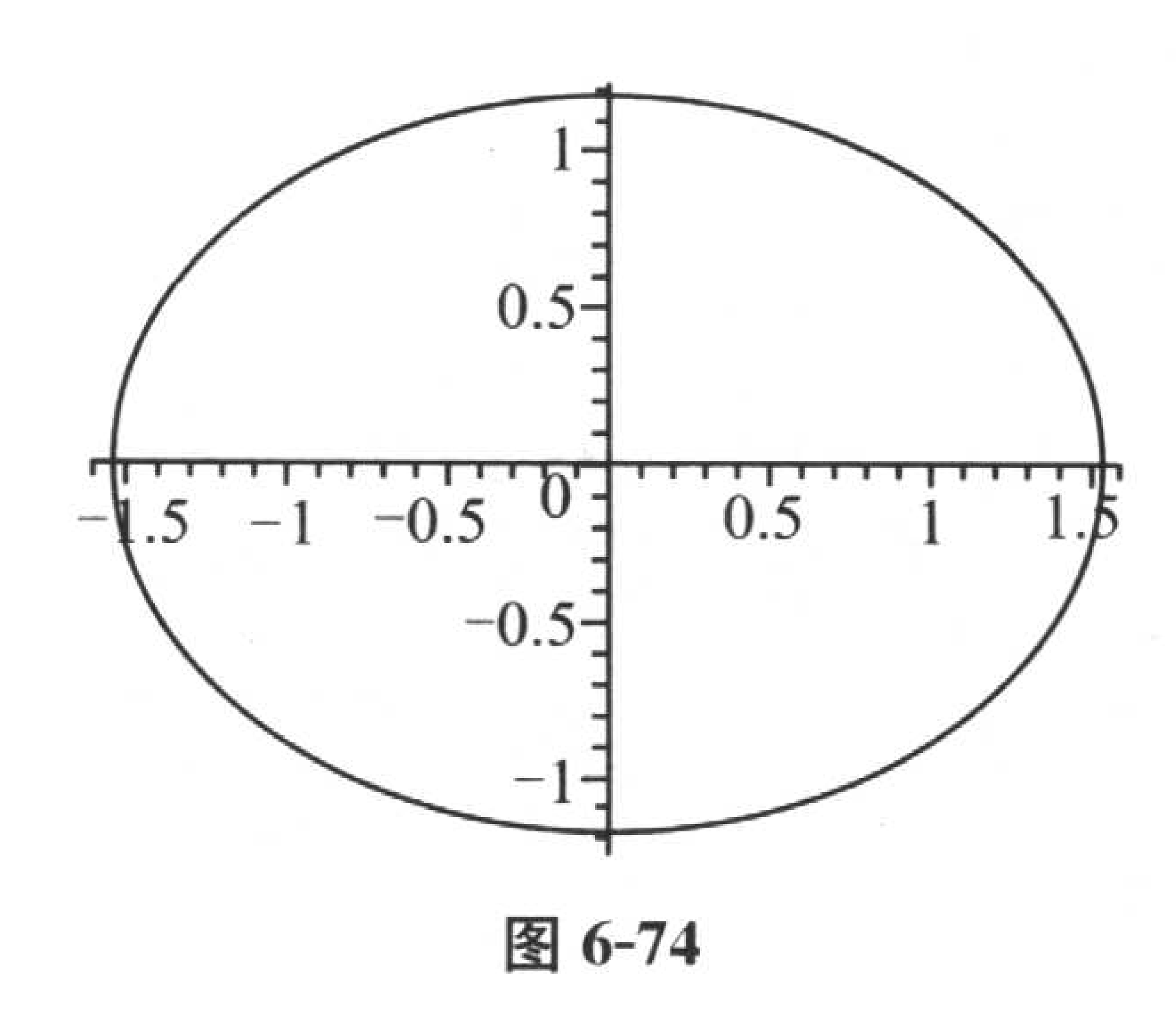


图 6-74

> complexplot3d(sec(z),z=-3-3*I..3+3*I,axes=framed);

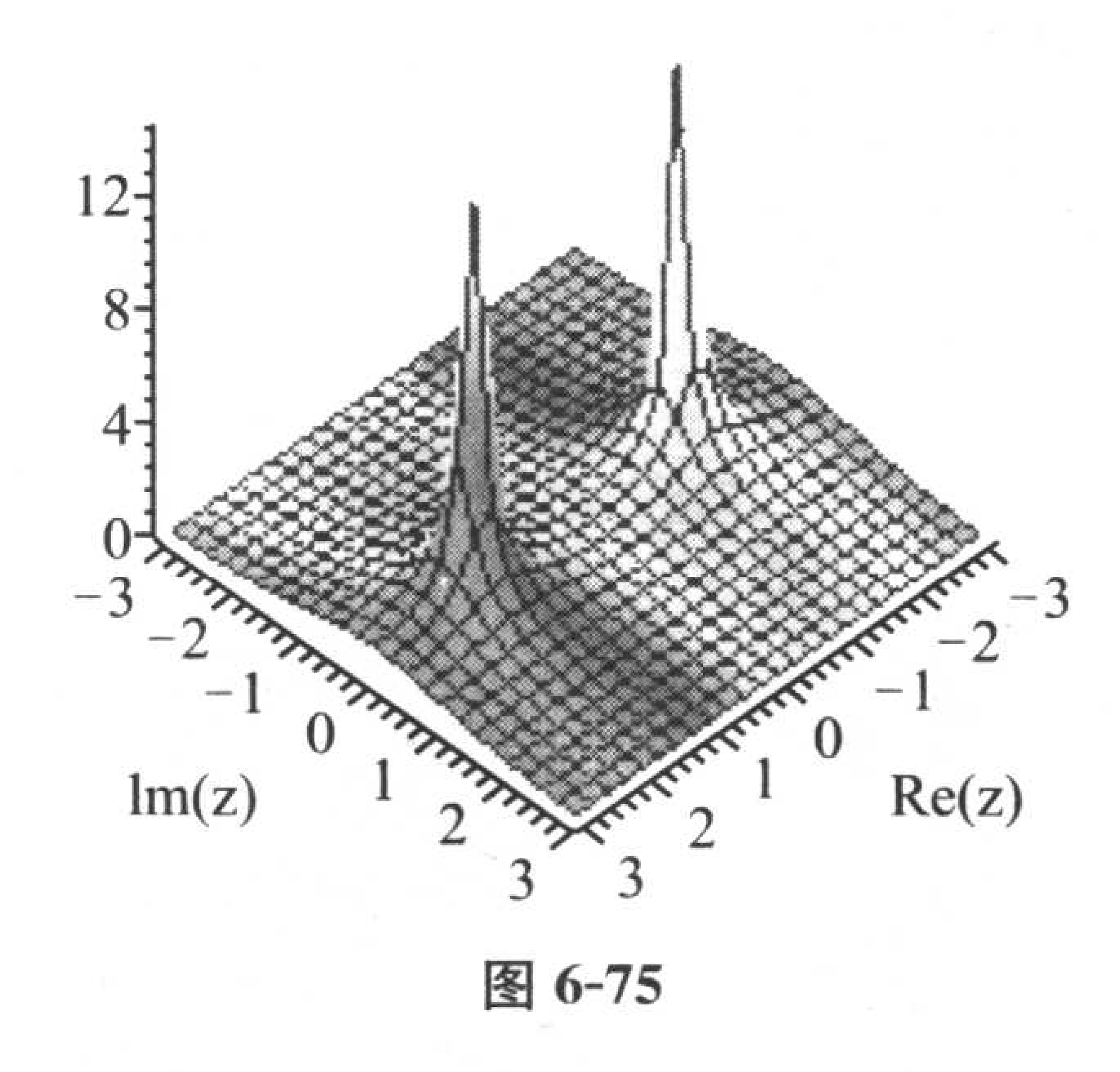


图 6-75

Conformal 和 conformal3d 命令的功能是:绘制复平面上的矩形区域在解析函数映射下的像。

【例 52】 共形映射(图 6-76~6-81)。

> conformal(exp(z),z=1..Pi*I);

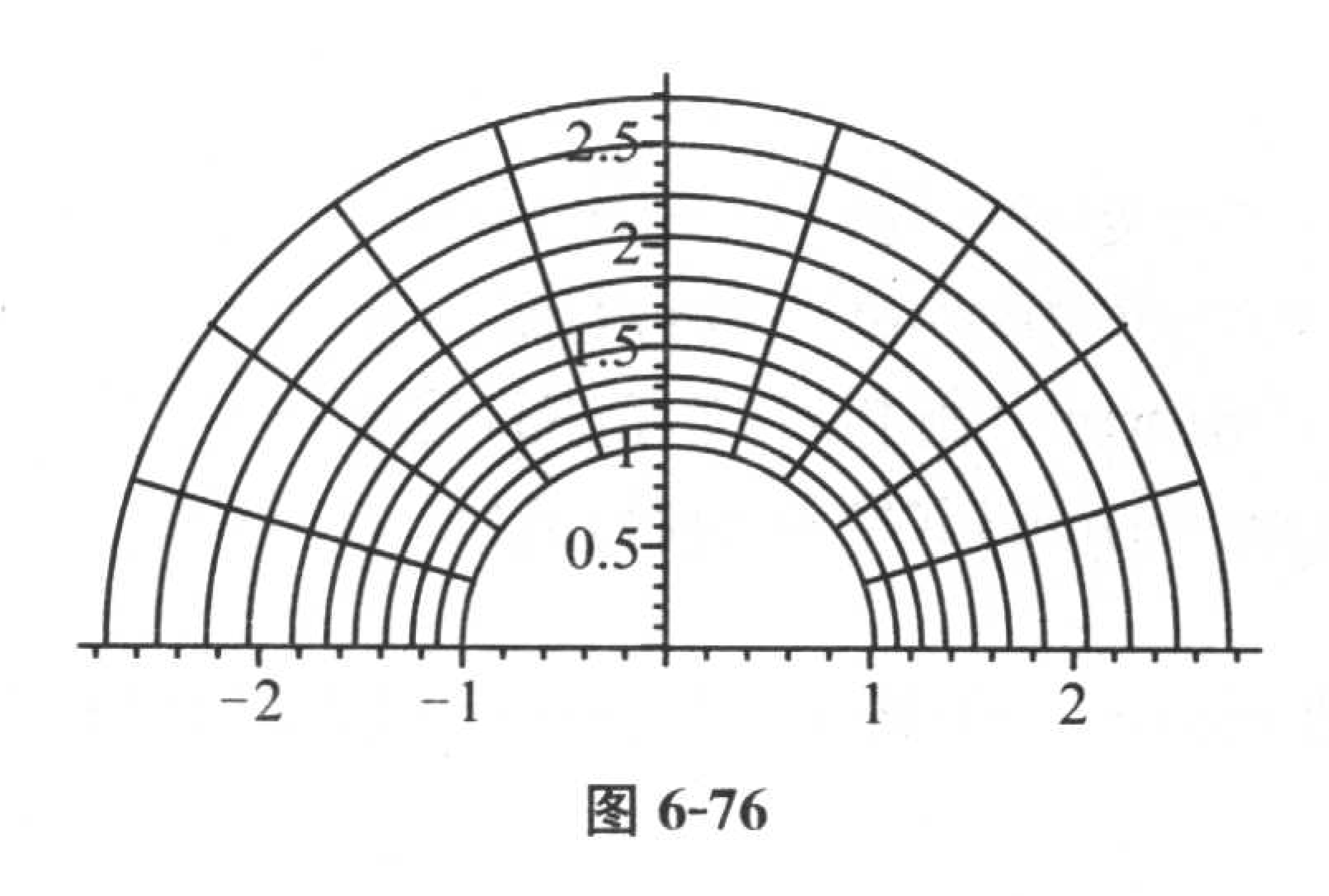


图 6-76

> conformal(z^3,z=-1-I..1+I);

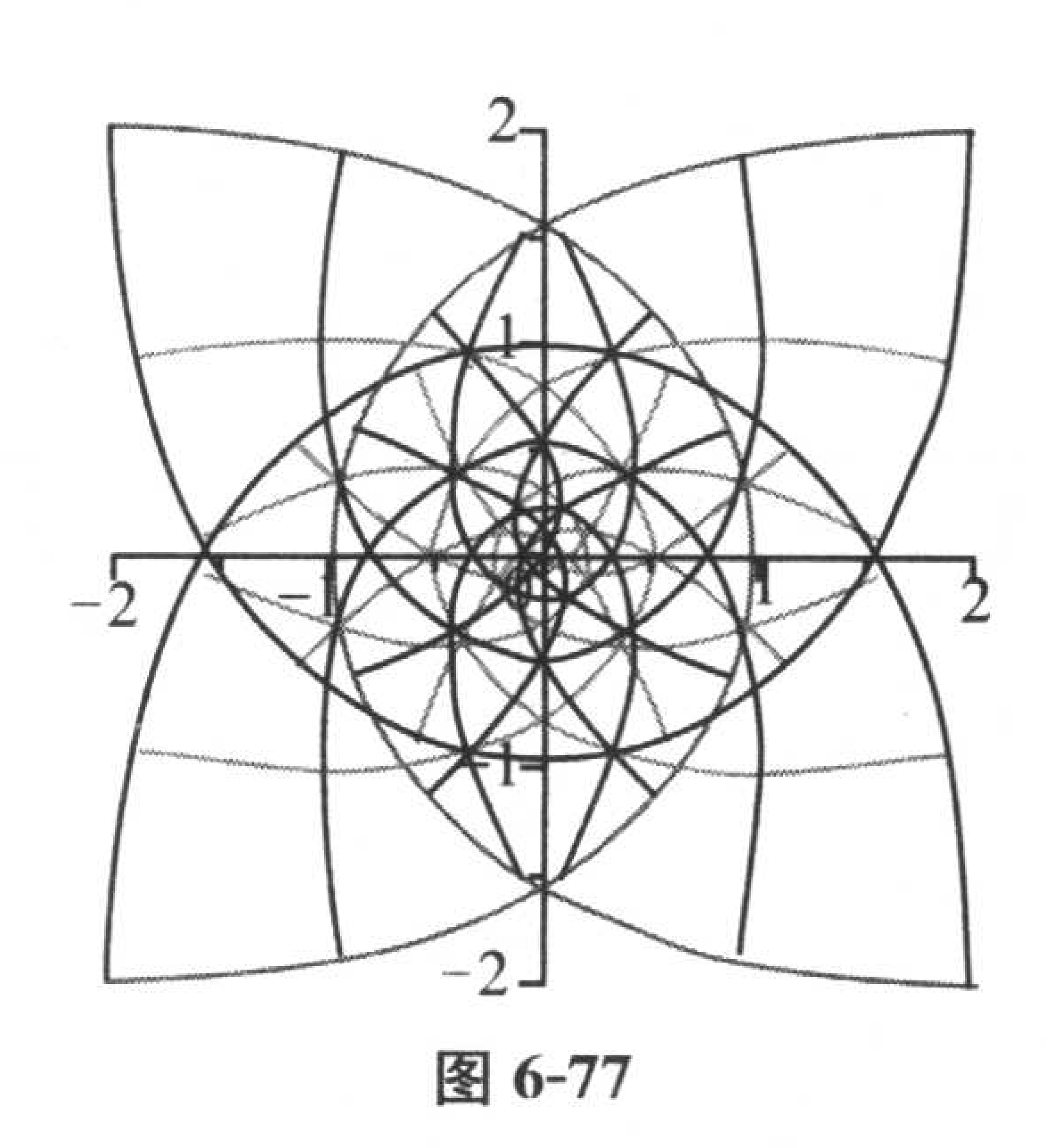


图 6-77

```
> conformal(1/z^3,z=-1-I..1+I,-6-5*I..6+5*I);
```

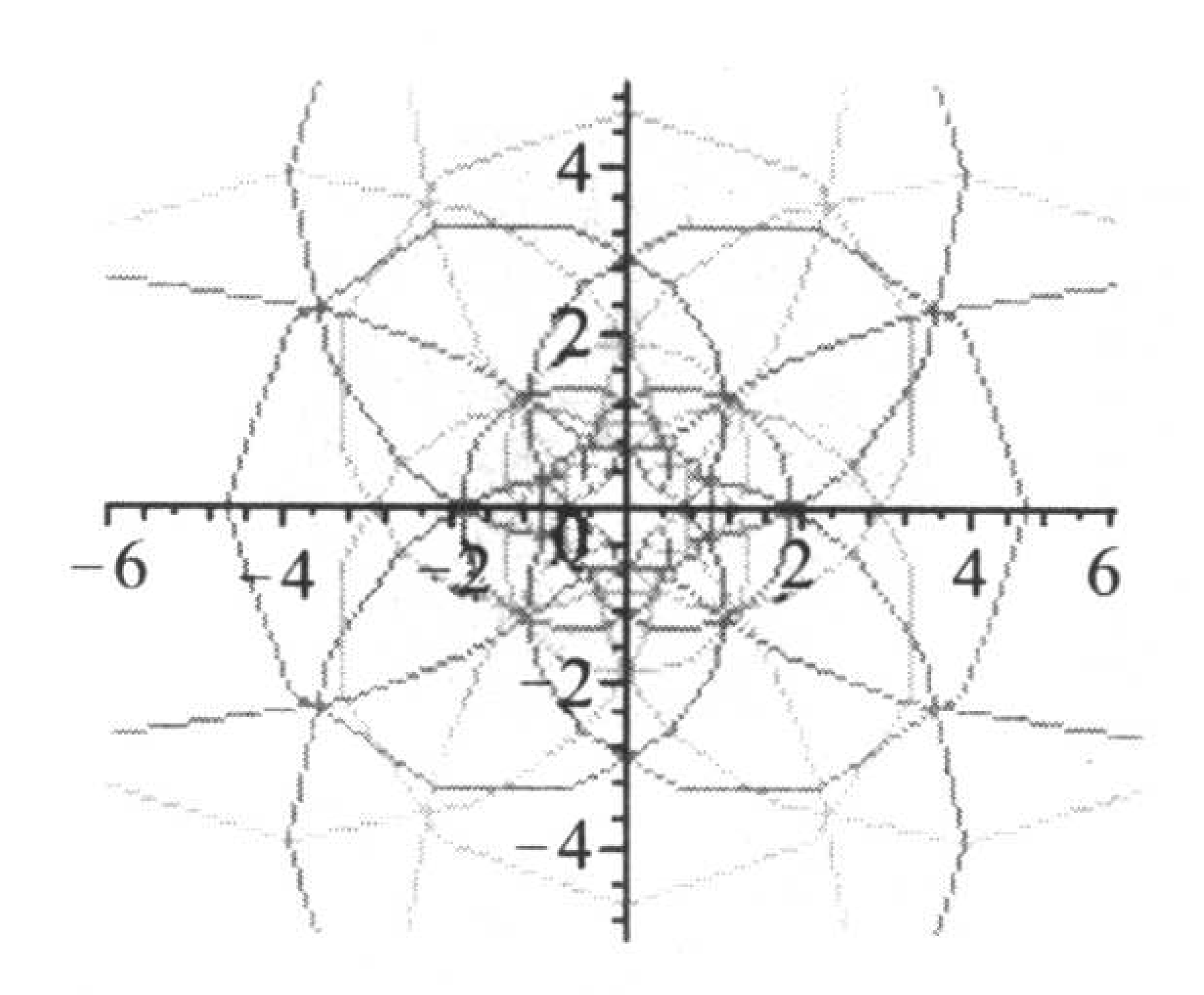


图 6-78

```
> conformal((z-1)/(z+1),z=-2-2*I..2+2*I,-3-3*I..3+3*I);
```

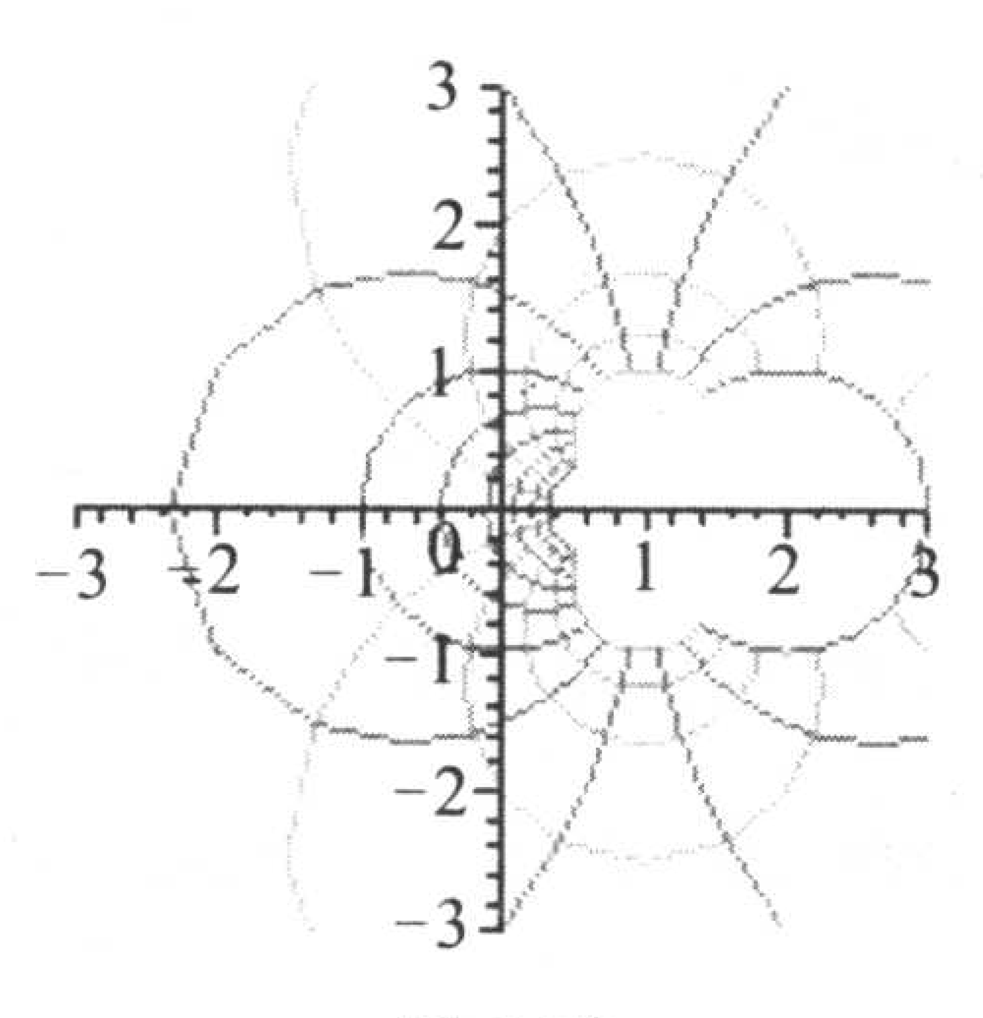


图 6-79

```
> conformal(cos(z),z=0..2*Pi+I*Pi);
```

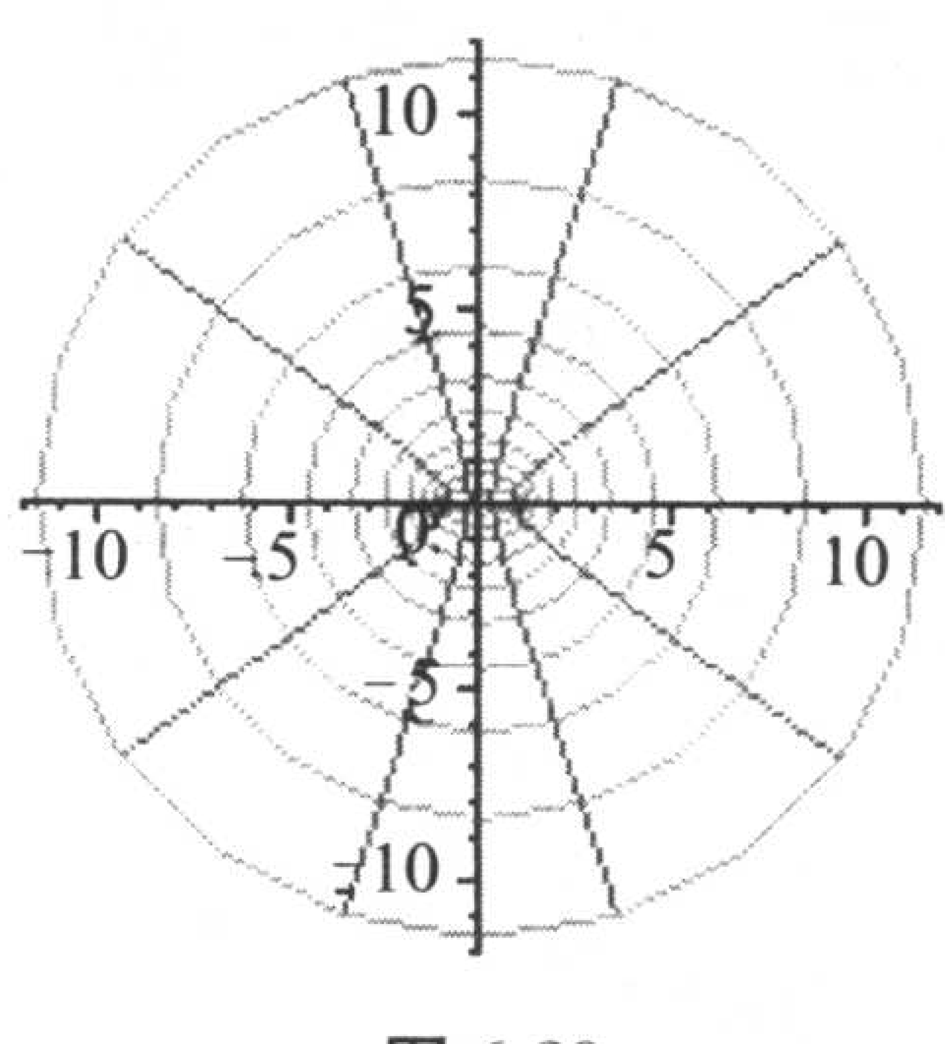


图 6-80

```
> conformal3d(cos(z),z=0..2*Pi+I*Pi);
```

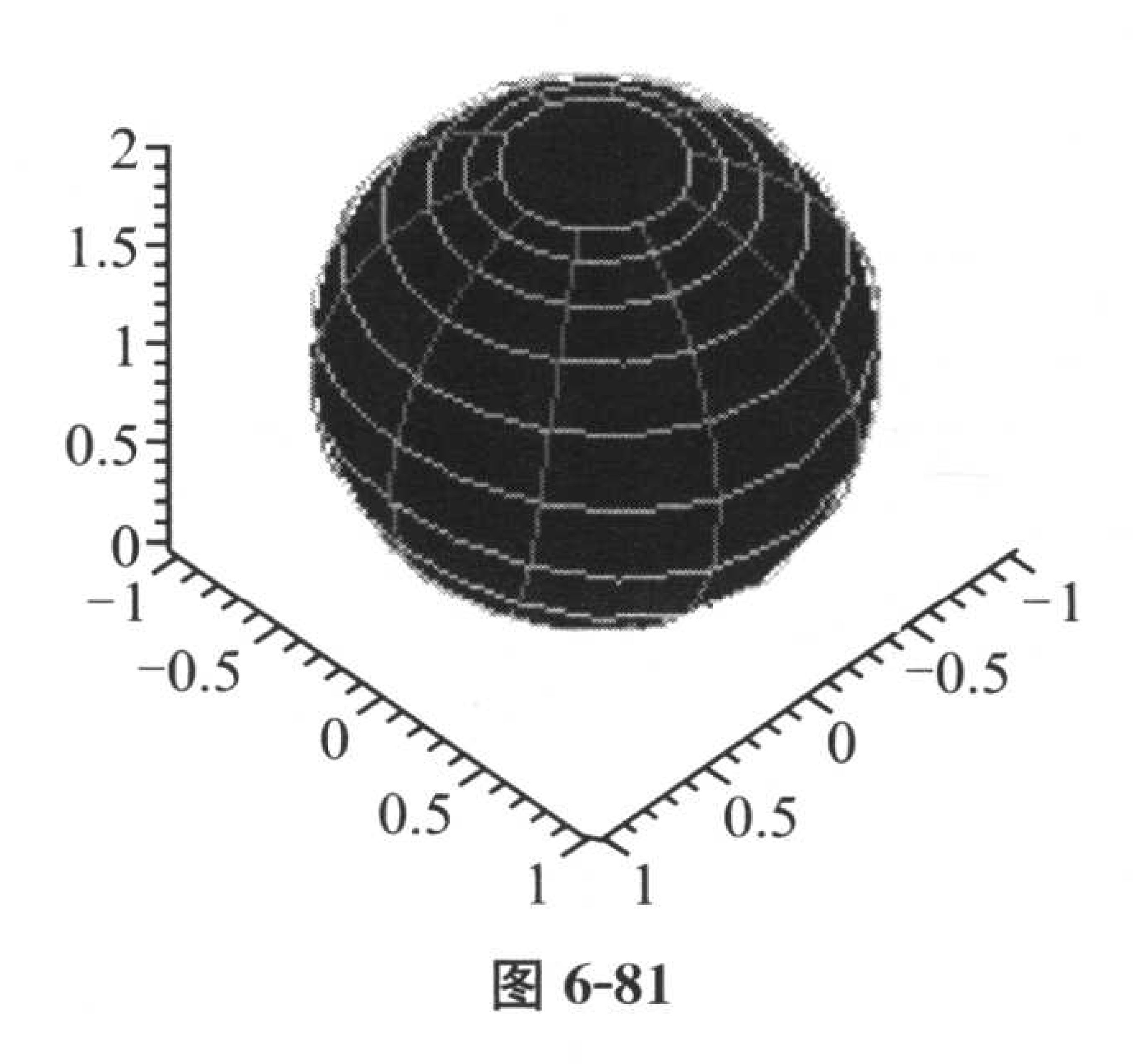


图 6-81

6.3.8 其他画图函数

1. 管状图

tubeplot(空间曲线,选项)

【例 53】 弹簧(图 6-82)、喇叭花(图 6-83)、双环(图 6-84)。

```
> with(plots):
> tubeplot([t,3*cos(t),3*sin(t)],t=-4*Pi..4*Pi,radius=1);
```

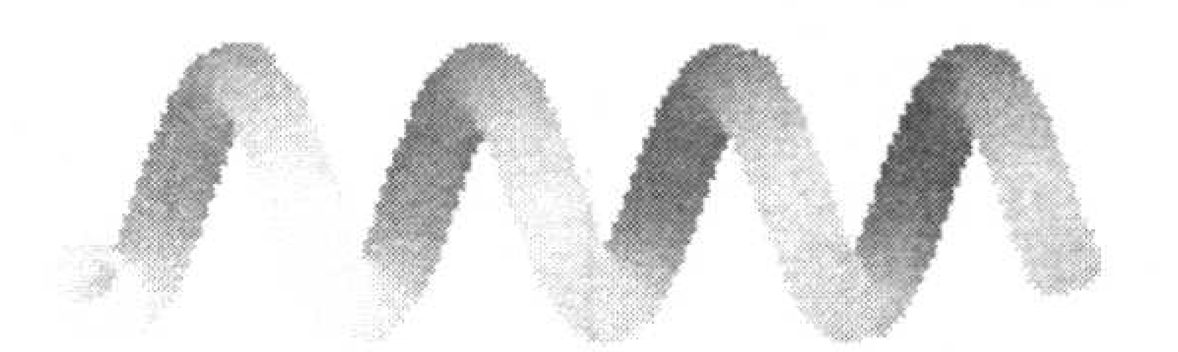


图 6-82

```
> tubeplot({[t,cos(t),sin(t)],[t,cos(t),-sin(t)]},t=-1..1, radius=t);
```

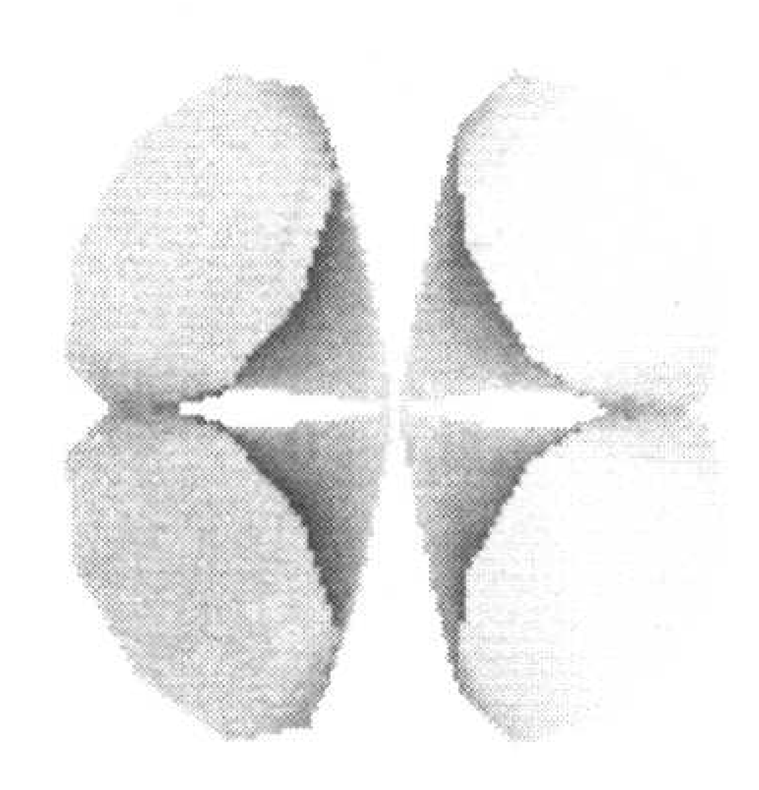


图 6-83

```
> tubeplot({[cos(t),sin(t),0],[0,sin(t)-1,cos(t)]},t=0..2*Pi, radius=
1/4);
```

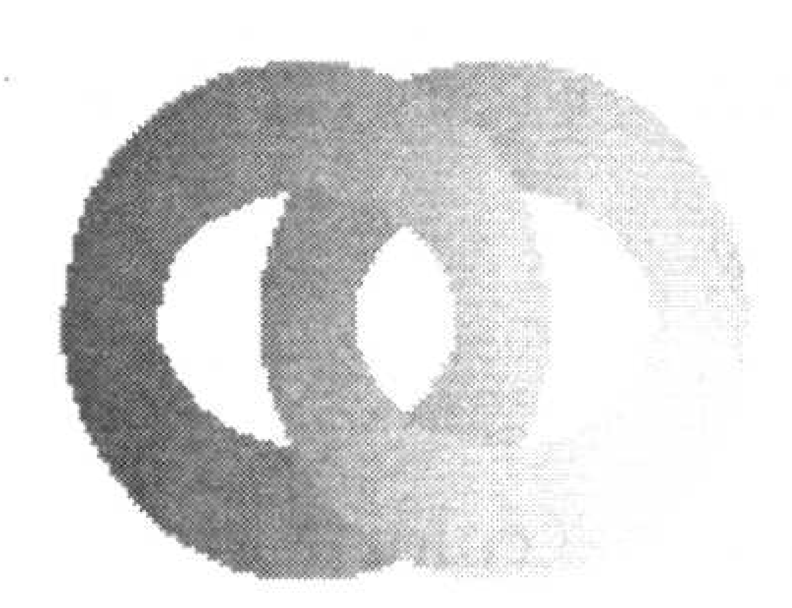


图 6-84

2. 箭头或向量

arrow(向量,选项)

【例 54】 箭头图形(图 6-85,6-86)。

```
> a1:=arrow([seq(<sin((2*i-1)/6*Pi),cos((2*i-1)/6*Pi)>,i=1..6)],
  color=red,shape=arrow):
  a2:=arrow([seq(<sin(2*i/6*Pi),cos(2*i/6*Pi)>,i=1..6)],
  color=blue,axes=boxed):
> display(a1,a2);
```

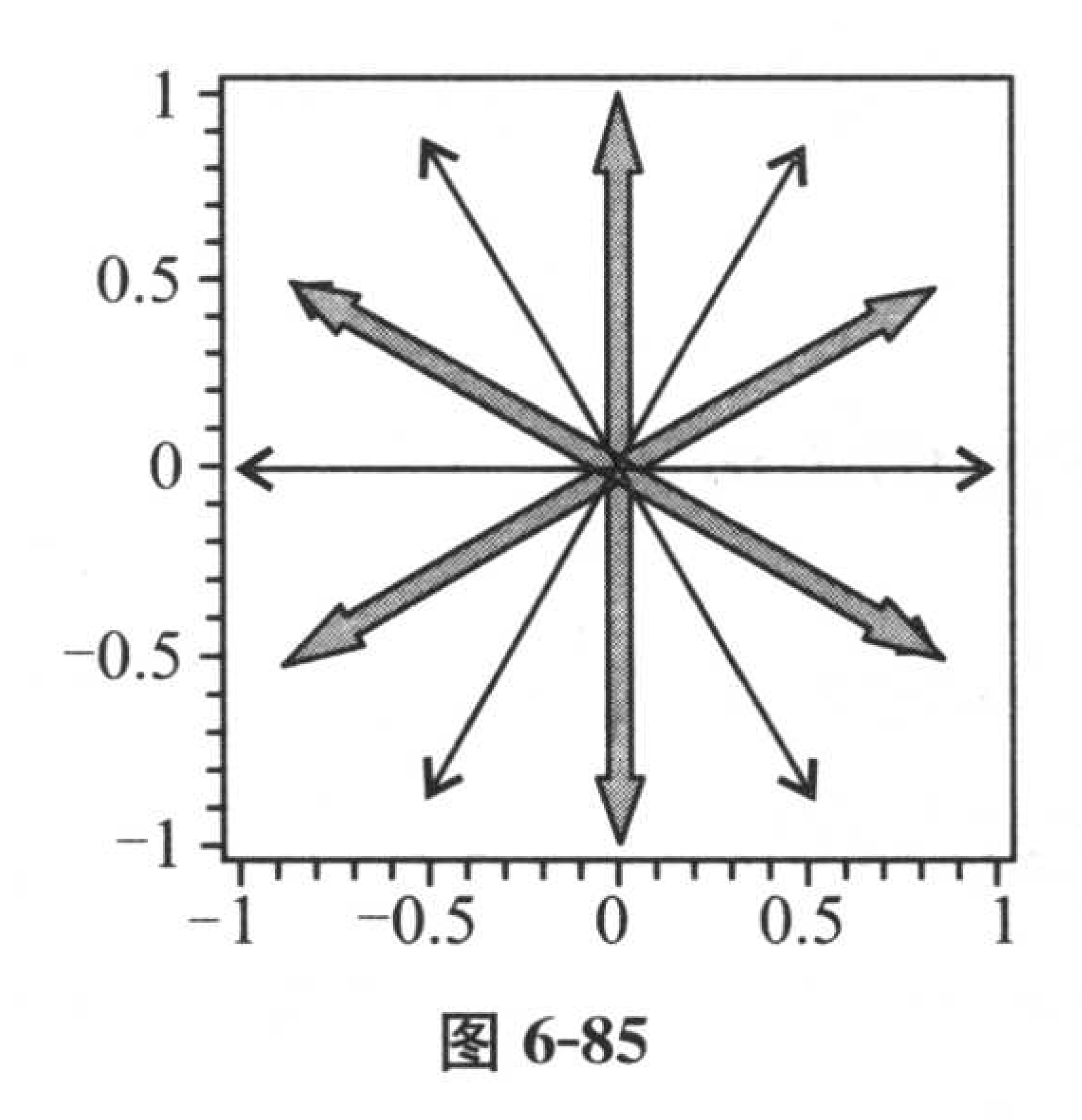


图 6-85

```
> arrow([seq(<cos(i/3*Pi),sin(i/3*Pi),1>,i=1..6)]);
```

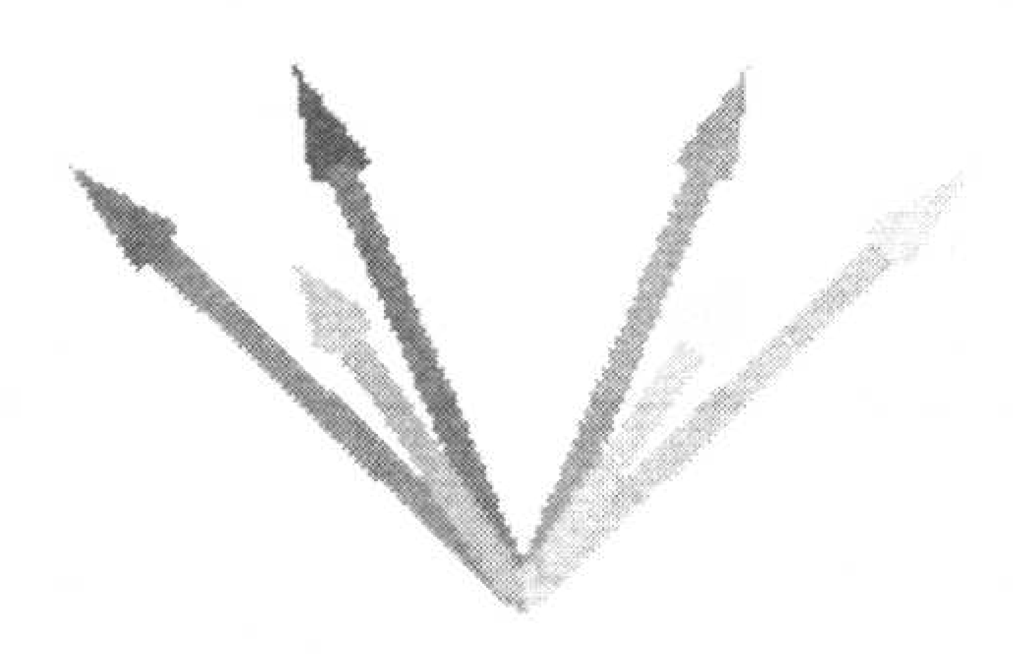


图 6-86

3. 不等式区域

inequal(不等式,范围,选项)

【例 55】 求解不等式(图 6-87)。

```
> inequal({x+y>0,x-y<=1},x=-3..3,
  y=-3..3, optionsexcluded=(color=gray)); # 蓝色区域和实线边界为不等式的解集
```

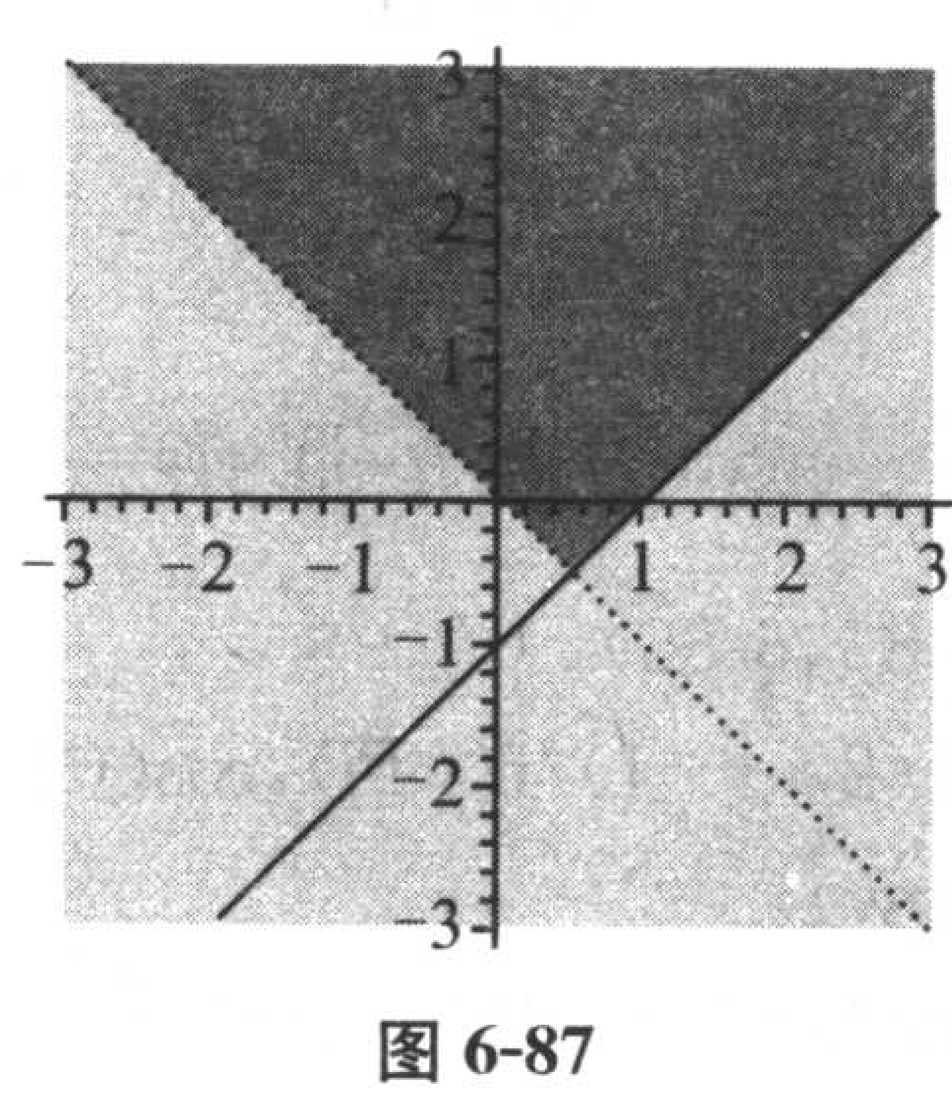


图 6-87

6.4 动画演示

6.4.1 动画命令(animate)

要$[a,b]$上的函数 $f(x)$动起来,作动画演示,我们可以使用命令:

animate(f(x,t),x=a..b,t=c..d)

animate 函数在 plots 函数包中,使用前要先调出 plots 函数包。

【例 56】 动画演示 sin(x)图像(图 6-88)。

```
> with(plots):
> animate(sin(x-t),x=-2*Pi..2*Pi,t=0..10,scaling=constrained);
```

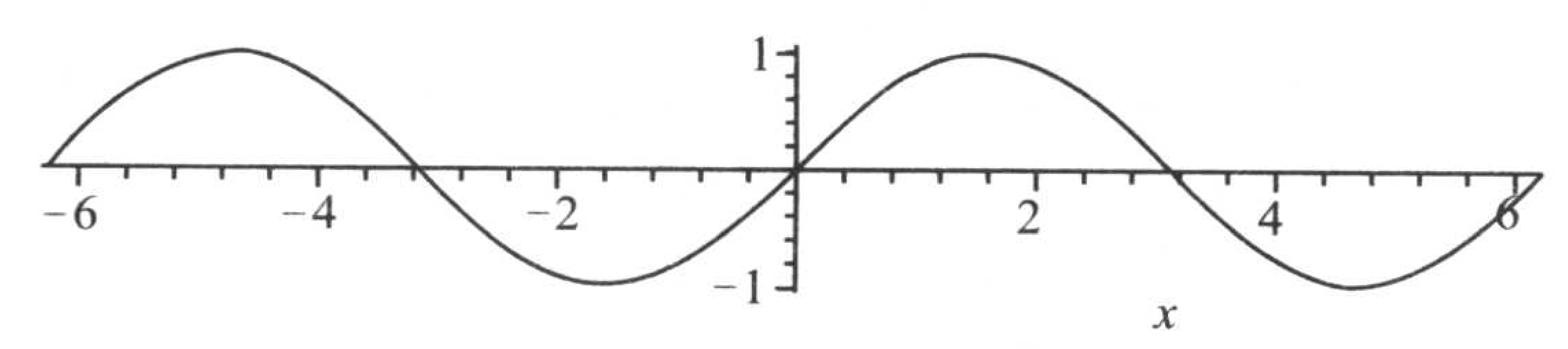


图 6-88

运行命令后，看到的是静止的图，为 t=0 时的函数图像。单击上下文工具栏（图 6-89）中的▷按钮，图形立即"动"了起来。工具栏中按钮的功能见图 6-90。

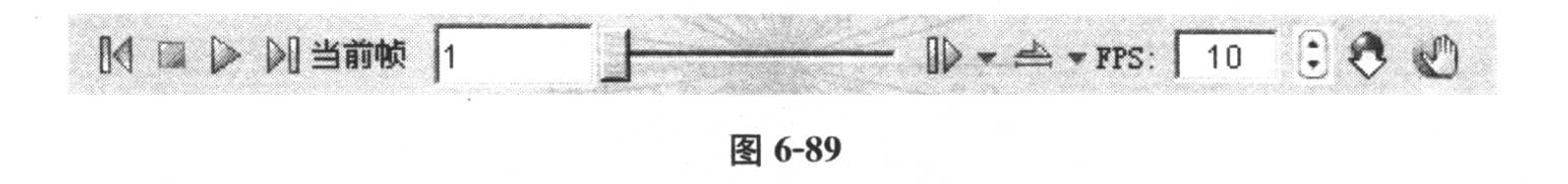


图 6-89

、、、	播放前一帧、停止/暂停动画、开始/继续播放动画、播放下一帧
当前帧 1	显示/设置当前帧数
、、FPS: 10	设置播放方向、播放次数、动画速度（每秒帧数）
	缩放绘图坐标轴、平移坐标轴

图 6-90

animate 函数的一般形式为：

animate(画图命令，函数，范围，选项)

如果画图命令是 plot，则可省略。

【例 57】　演示一条摆动的曲线(图 6-91)。选项 frames=value 可设置动画的帧数，默认值 16。系统通过依次显示每帧图形，从而演示动画过程。其他选项和 plot 函数的选项基本一致。

```
> animate(x^t,x=0..1,t=0..6,frames=40,color=blue,axes=none);
```

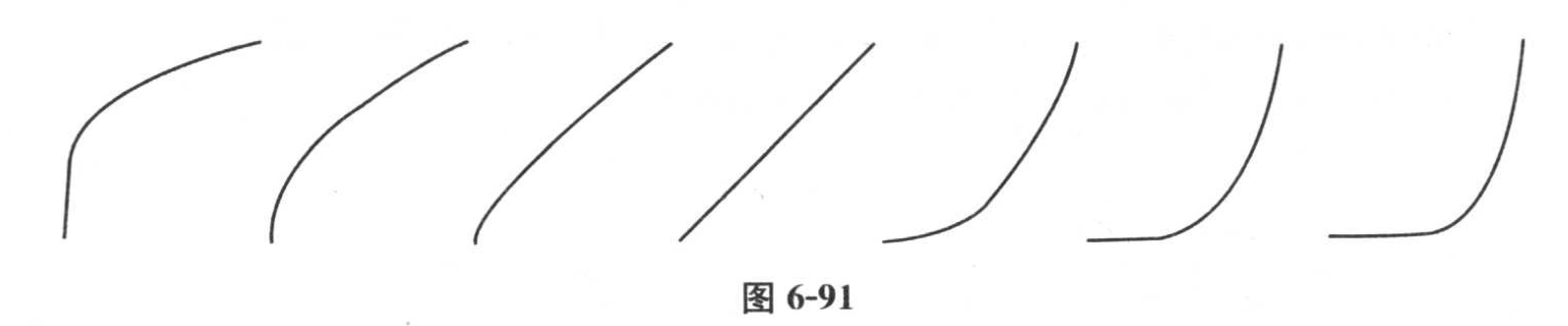


图 6-91

【例 58】　演示过程为参数的动画(图 6-92)。

```
> animate((x,t)->sin(t*x),0..Pi,1..16,coords=polar,numpoints=100);
```

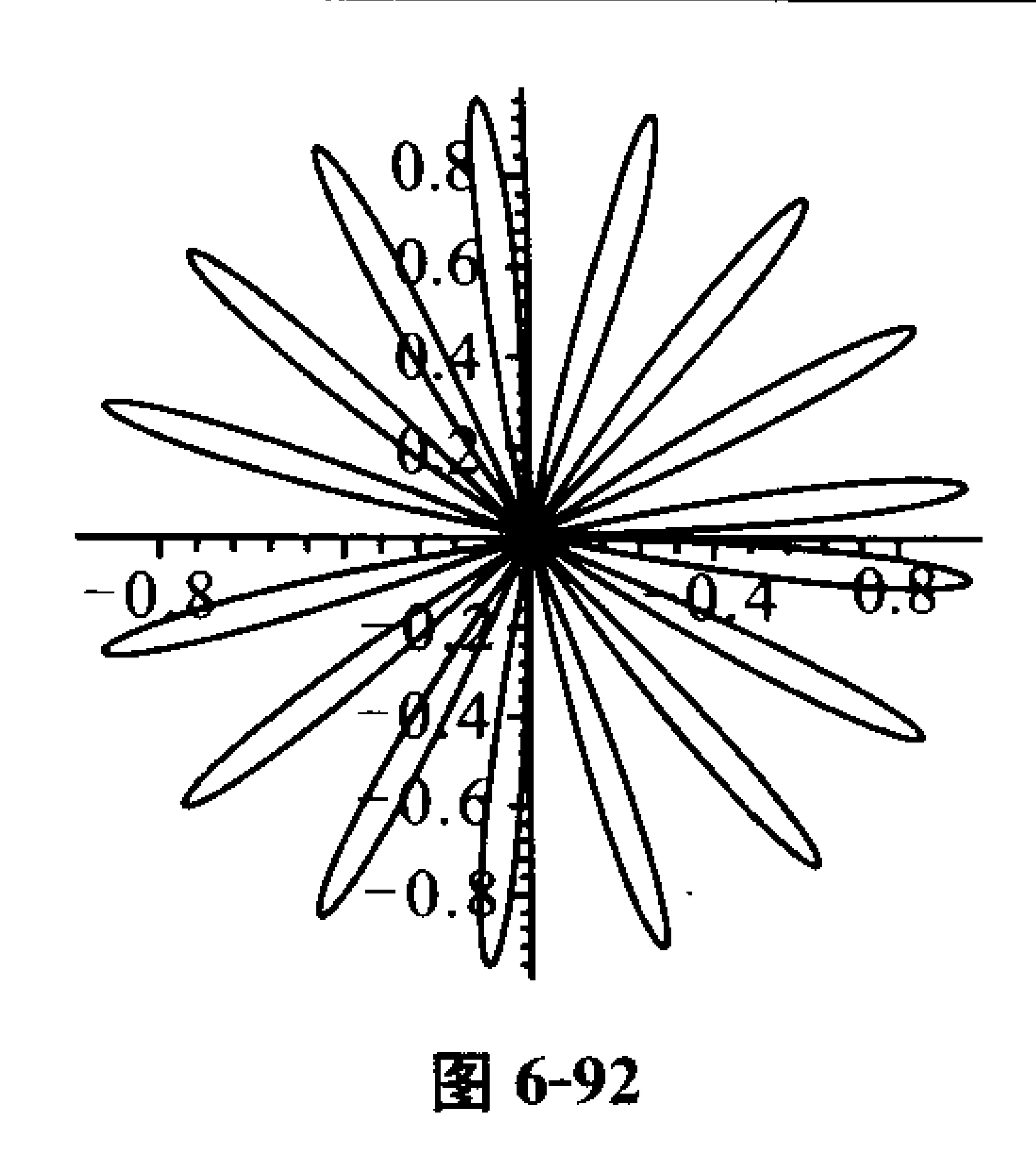


图 6-92

【例 59】 将圆$(x-1)^2+y^2=1$绕原点旋转(图 6-93)。

```
> animate([cos(s)+cos(t),sin(s)+sin(t),s=0..2*Pi],t=0..2*Pi);
```

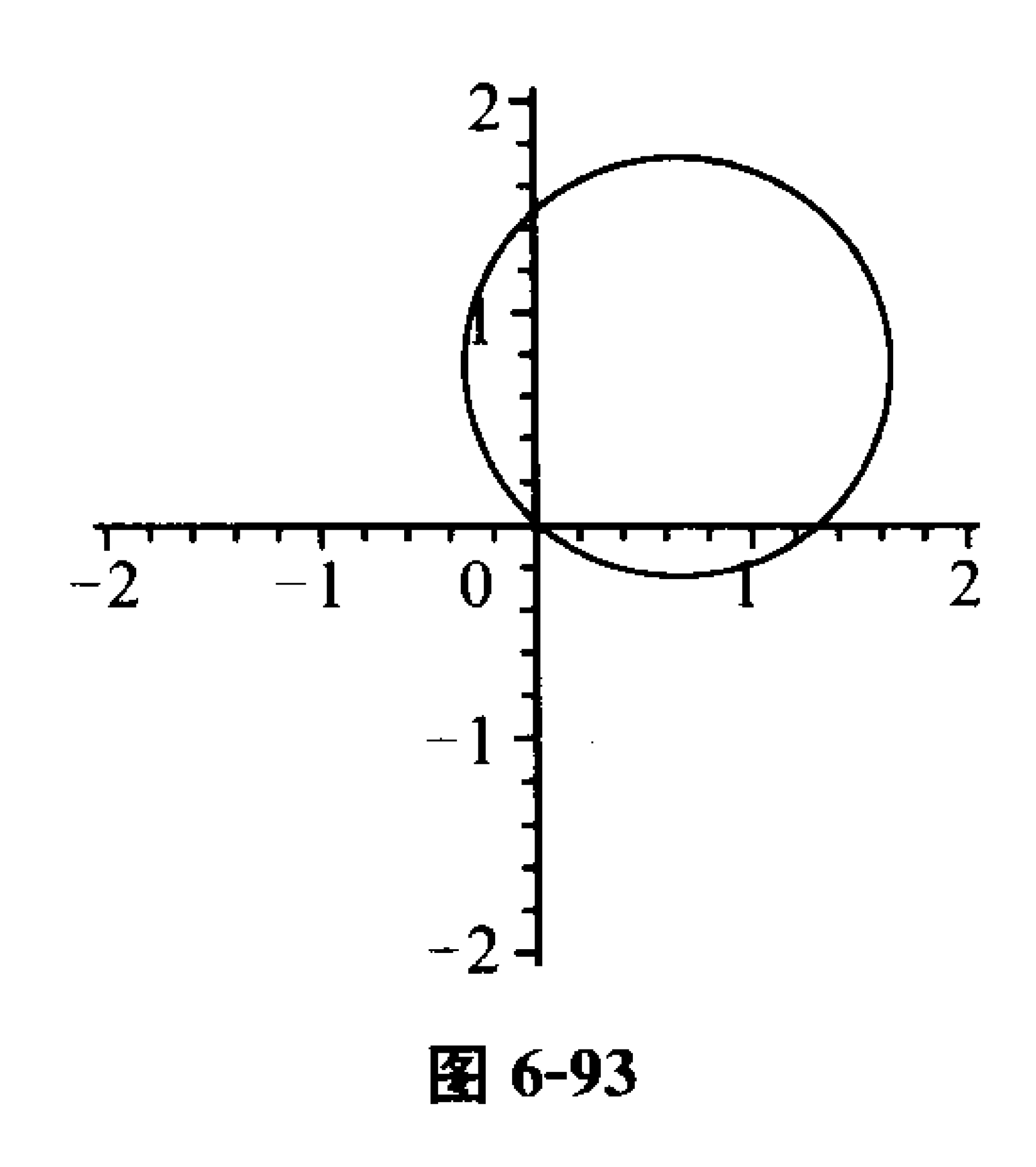


图 6-93

【例 60】 红点沿着蓝色曲线滑动(图 6-94)。

```
> curve:=plot(sin(x),x=0..10,color=blue):
> animate(pointplot,[[t,sin(t)],symbol=circle,symbolsize=20],
  t=0..10,color=red,background=curve);
```

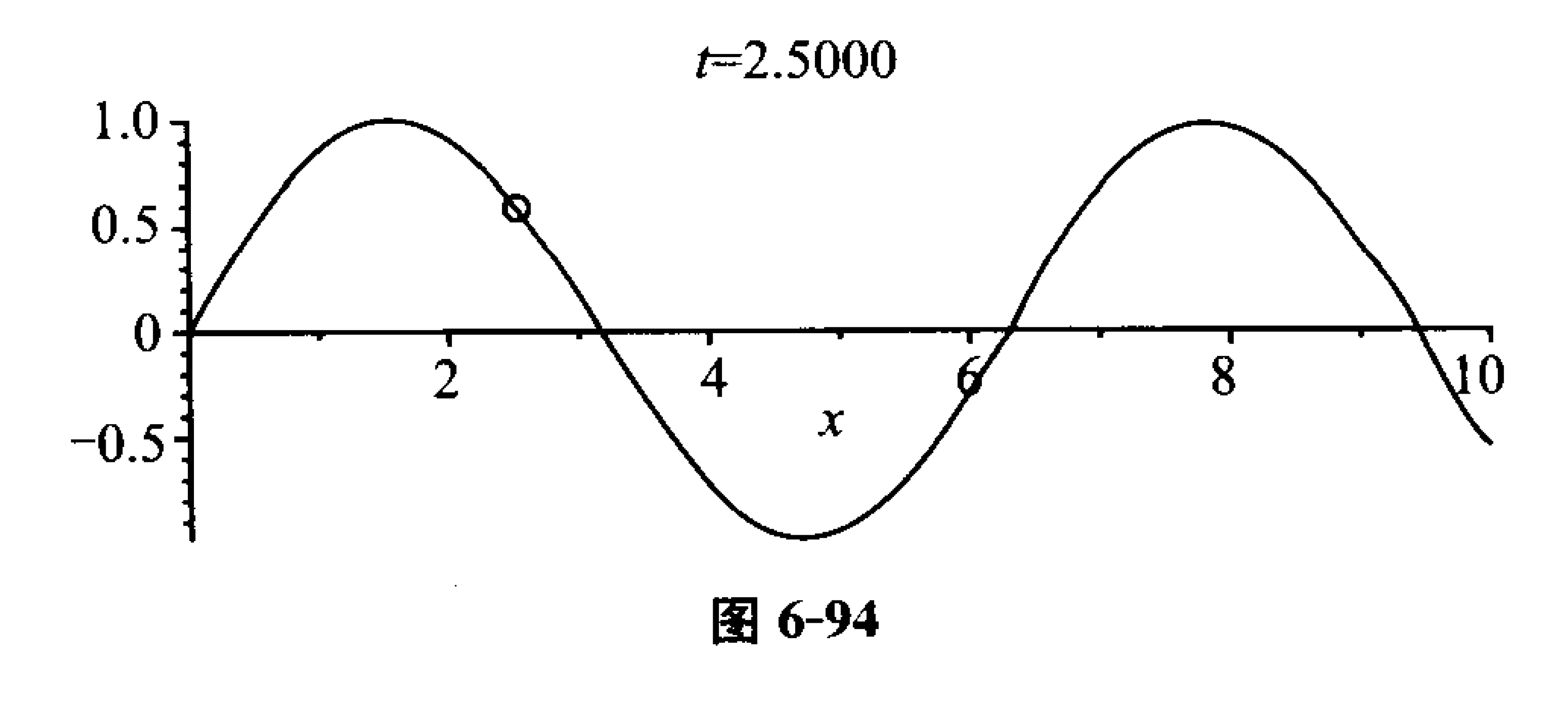


图 6-94

animate 也可用来演示三维动画。

【例 61】 直角坐标系下三维动画(图 6-95)。

> animate(plot3d,[x * sin(x * y+t)/2,x=-Pi..Pi,y=-4..4], t=-2 * Pi..2 * Pi);

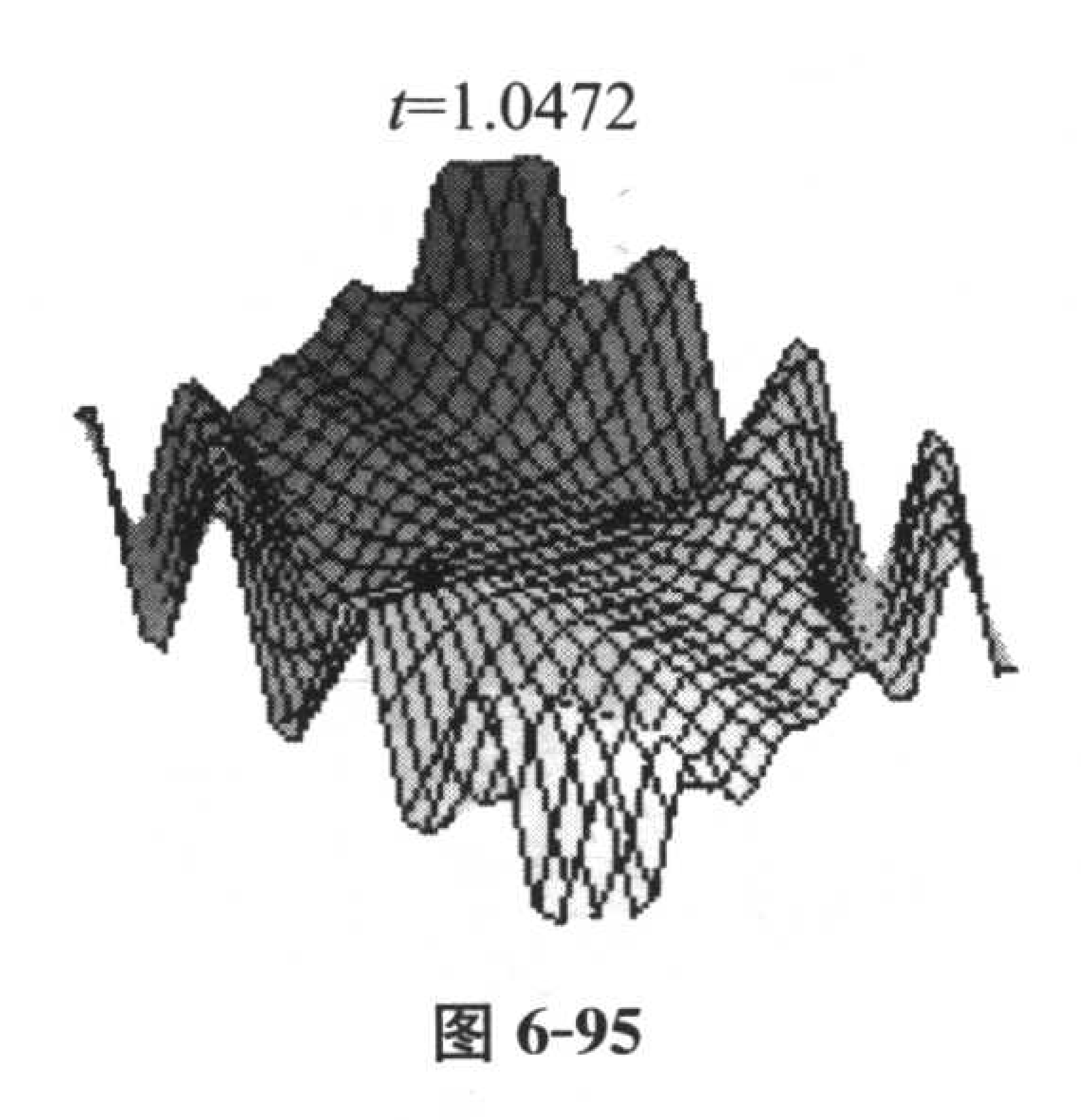


图 6-95

【例 62】 参数坐标下三维动画(图 6-96)。

> animate(plot3d,[cos(t * x * y),x=1..4,y=1..4],t=1..2);

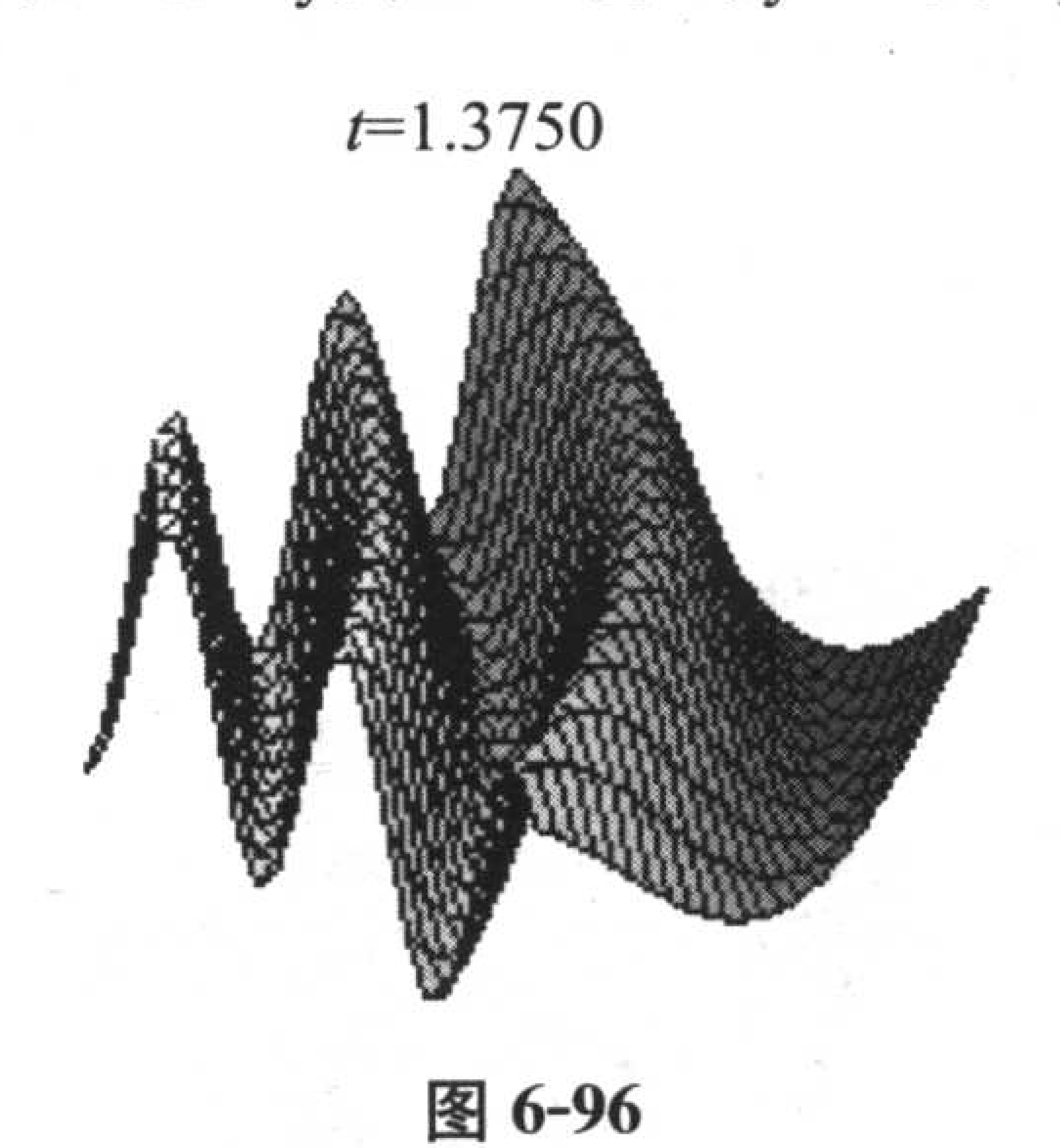


图 6-96

【例 63】 球坐标下三维动画(图 6-97)。

> animate(plot3d,[cos(t * x) * sin(t * y),x=-Pi..Pi,y=-Pi..Pi,
color=cos(x * y),coords=spherical, t=1..2]);

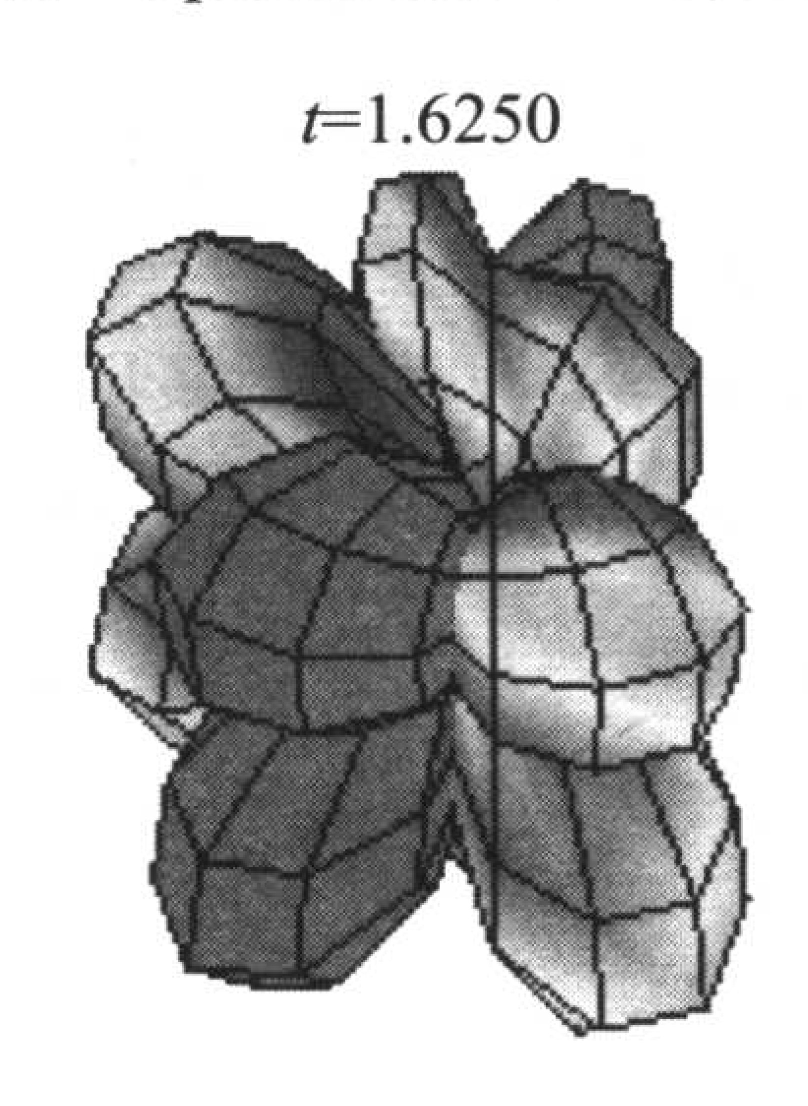


图 6-97

【例 64】 显示两个函数的三维动画(图 6-98)。

```
> animate(plot3d,[{(x+y)+t*x,(x-y)+t*y},x=-1..1,y=-1..1,
  coords=spherical], t=-4..4);
```

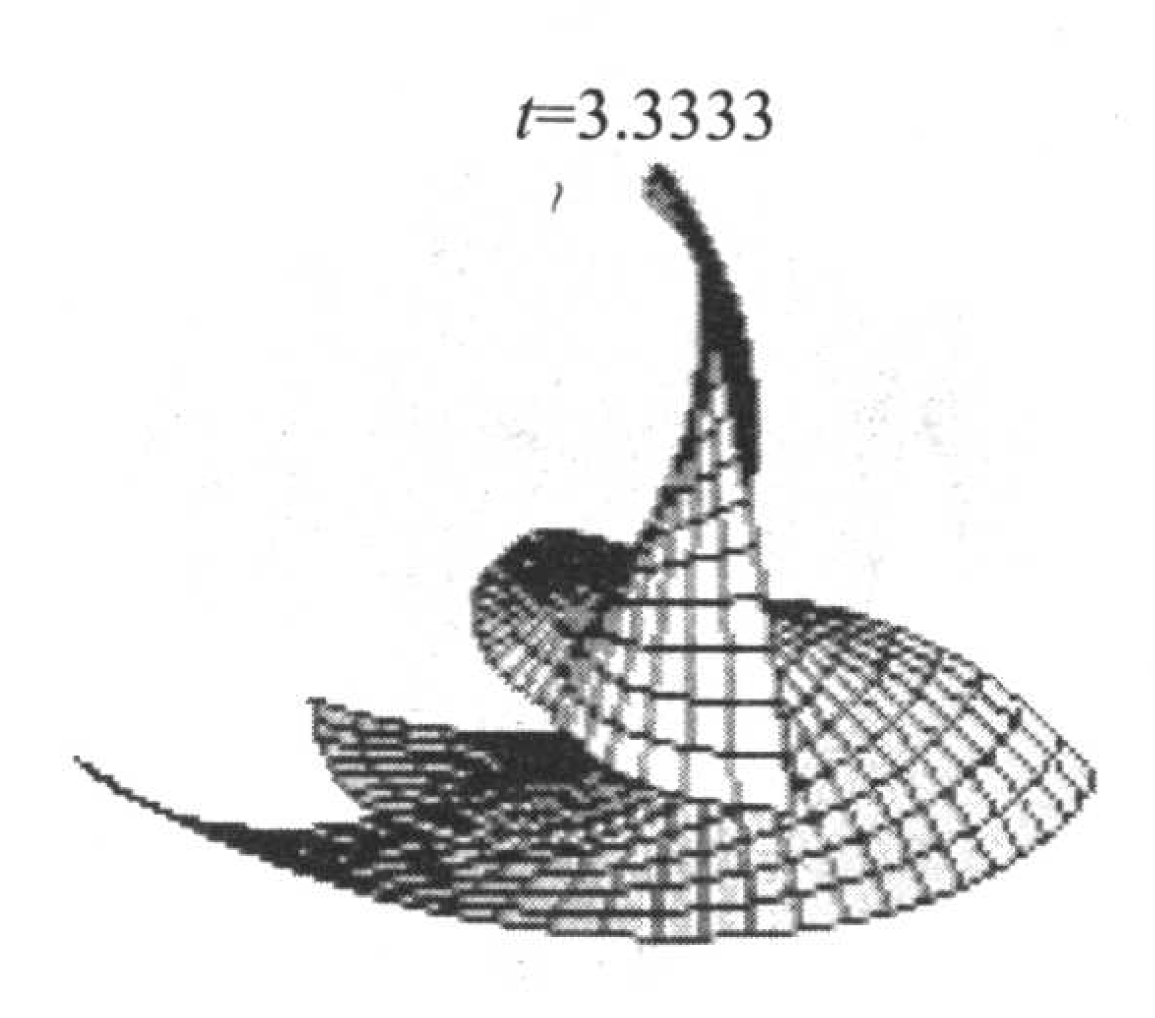


图 6-98

【例 65】 用函数图形作动画背景(图 6-99,6-100)。

```
> fg:=animate(plot3d,[sin(x*y*t),x=-Pi..Pi,y=-Pi..Pi],t=-1..
  1):
  bg:=plot3d(x-y,x=-Pi..Pi,y=-Pi..Pi): display(fg,bg);
```

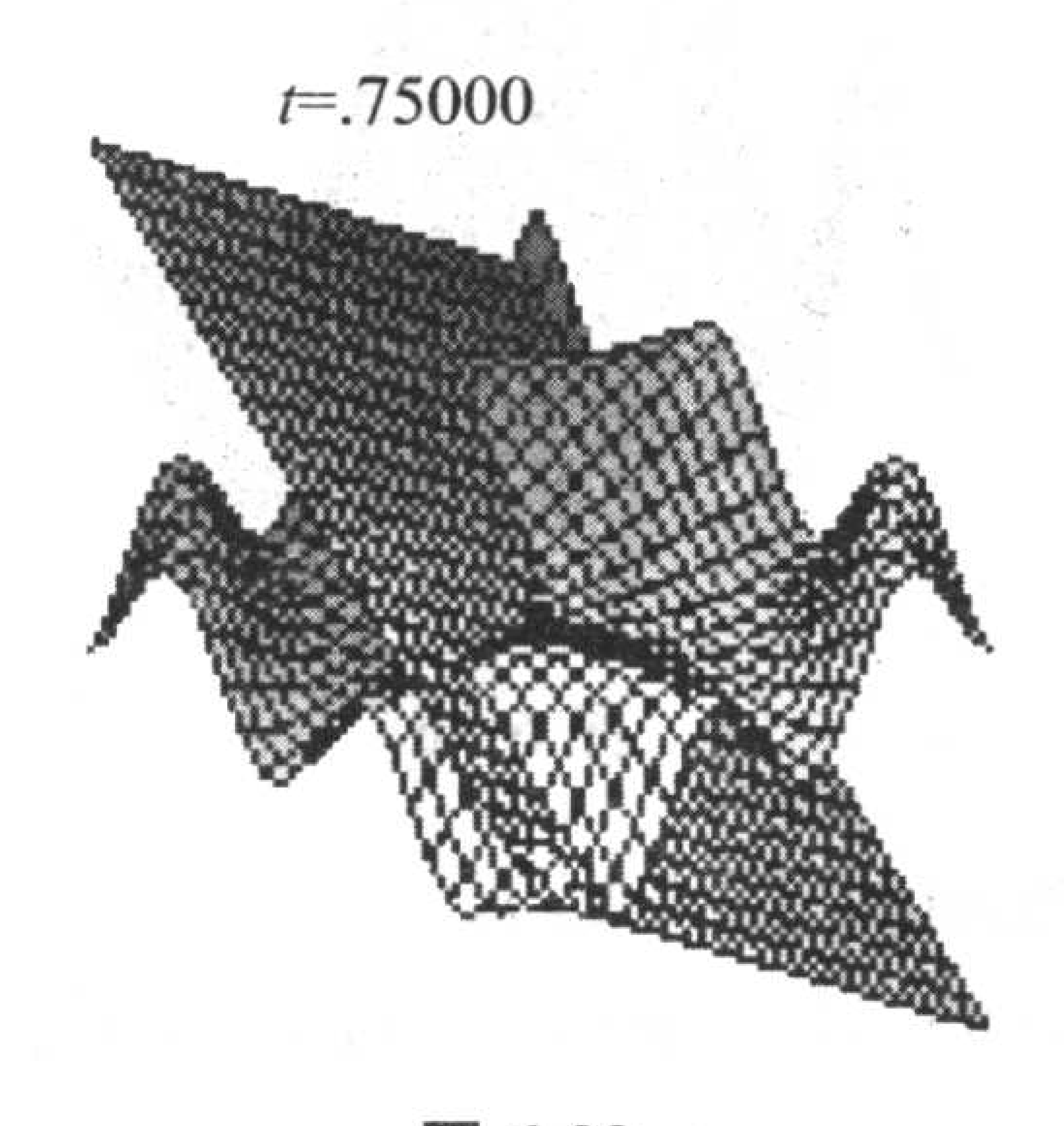


图 6-99

```
> g1:=animate(plot3d,[cos(t*x)*sin(t*y),x=-Pi..2*Pi,y=-Pi..2
  *Pi], t=1..2):
  g2:=plot3d(1.3^x*sin(y),x=-1..2*Pi,y=0..Pi,coords=spherical):
  g3:=plot3d(binomial,0..5,0..5): display(g1,g2,g3);
```

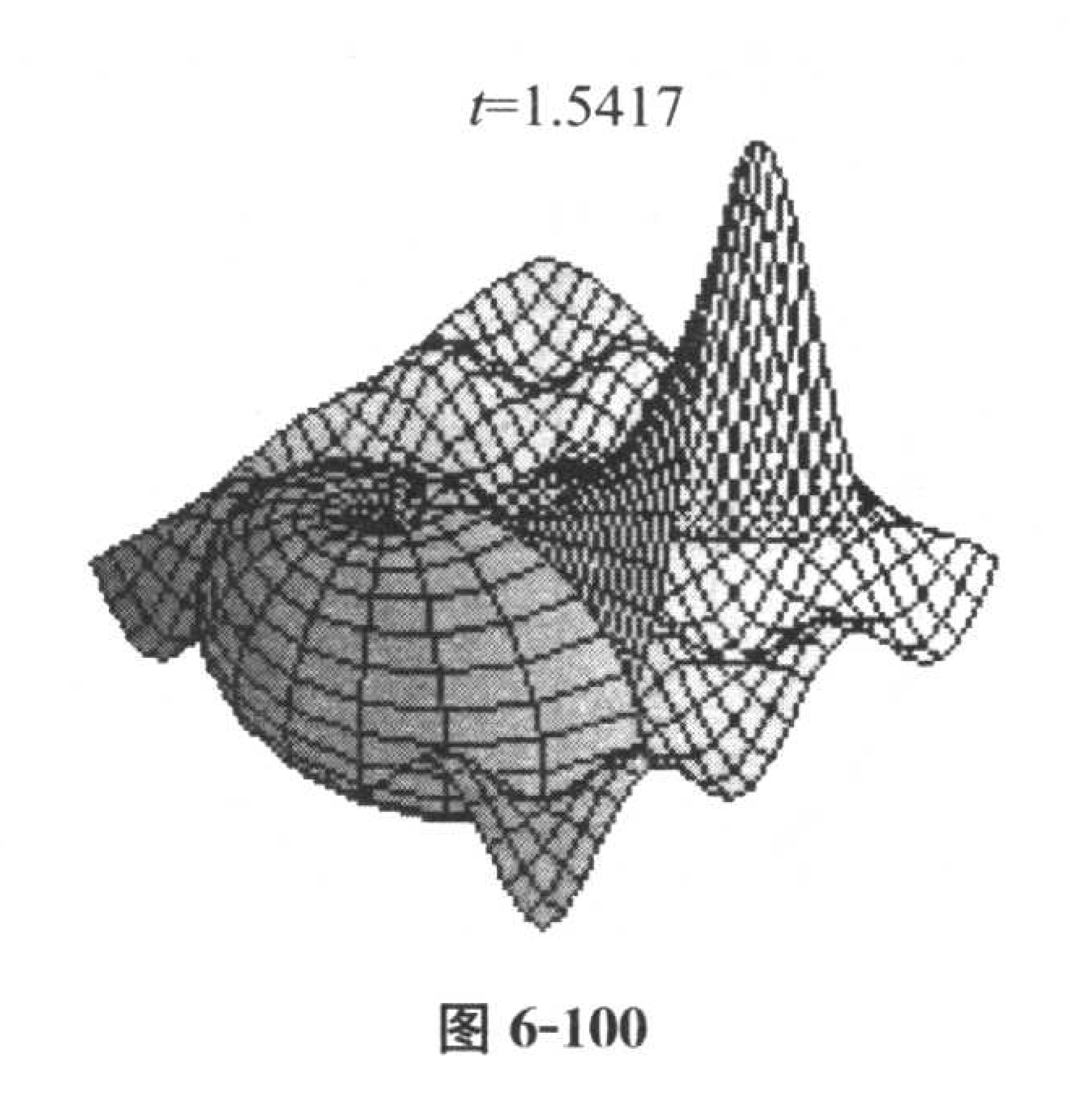


图 6-100

6.4.2　三维动画命令(animate3d)

在 maple 的以前版本中，animate3d 命令被专门用来显示三维动画，现在其功能已被 animate 所涵盖。在上一节中，我们已经给出了用 animate 和 plot3d 命令演示三维动画的例子。为此，用户只需掌握 animate 命令的各种用法即可。对 animate3d 的用法，我们下面仅举两例。在使用 animate3d 命令前，我们需要调用函数包 plots。

【例 66】　直角坐标系下三维动画。

```
> animate3d(x*sin(x*y+t)/2,x=-Pi..Pi,y=-4..4,t=-2*Pi..
  2*Pi);
```

输出略，与例 61 有何不同？

【例 67】　球坐标系下三维动画。

```
> animate3d(cos(t*x)*sin(t*y),x=-Pi..Pi,y=-Pi..Pi,t=1..2, \
  color=cos(x*y),coords=spherical);
```

输出略，与例 63 有何不同？

6.4.3　二维函数轨迹命令(animatecurve)

animatecurve 生成随自变量变化的曲线轨迹，它的一般用法为：

animatecurve(函数，范围，选项)

animatecurve 函数在 plots 函数包中，使用前要先调出 plots 函数包。

【例 68】　显示极坐标下点的轨迹(图 6-101)。

```
> with (plots):
```

> animatecurve(x,x=0..8 * Pi,coords=polar,numpoints=100);

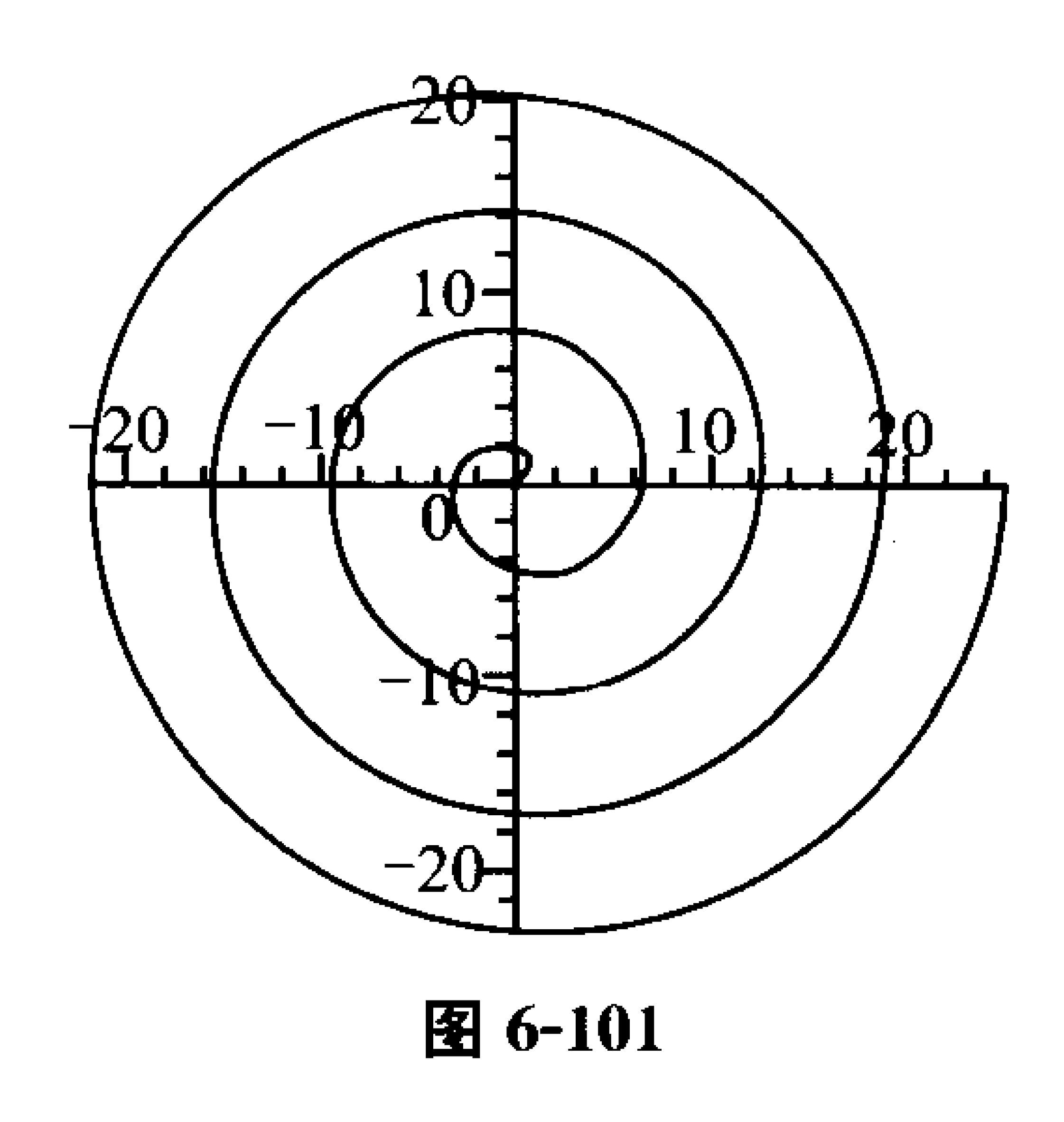


图 6-101

【例 69】 同时显示两个函数点的轨迹(图 6-102)。

> animatecurve({cos,sin},-Pi..Pi);

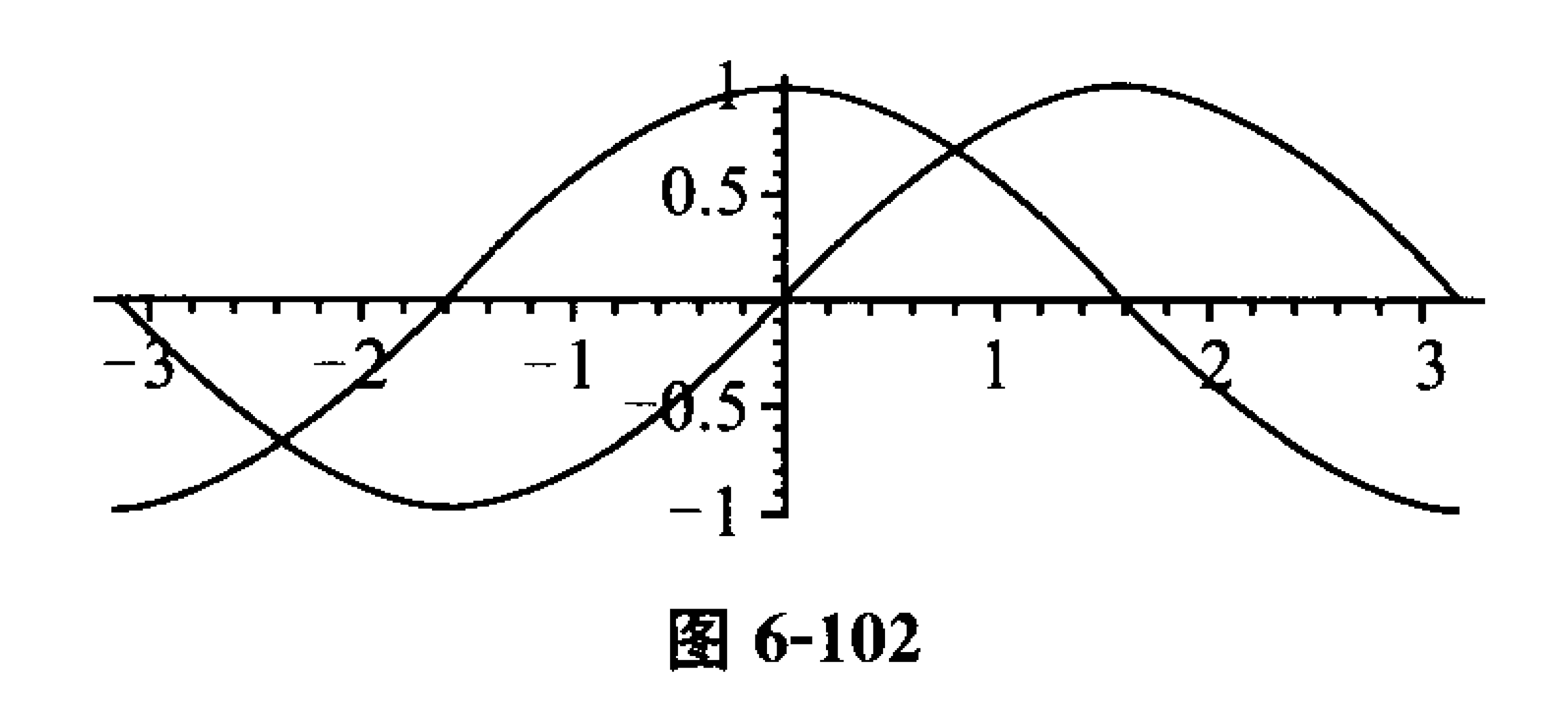


图 6-102

【例 70】 显示函数 $f(x)=\begin{cases}\sqrt{x}, x>0\\ x^2, x\leqslant 0\end{cases}$ 在[-2,2]点的轨迹(图 6-103)。

> animatecurve(x-> piecewise(x> 0,sqrt(x),x^2),-2..2);

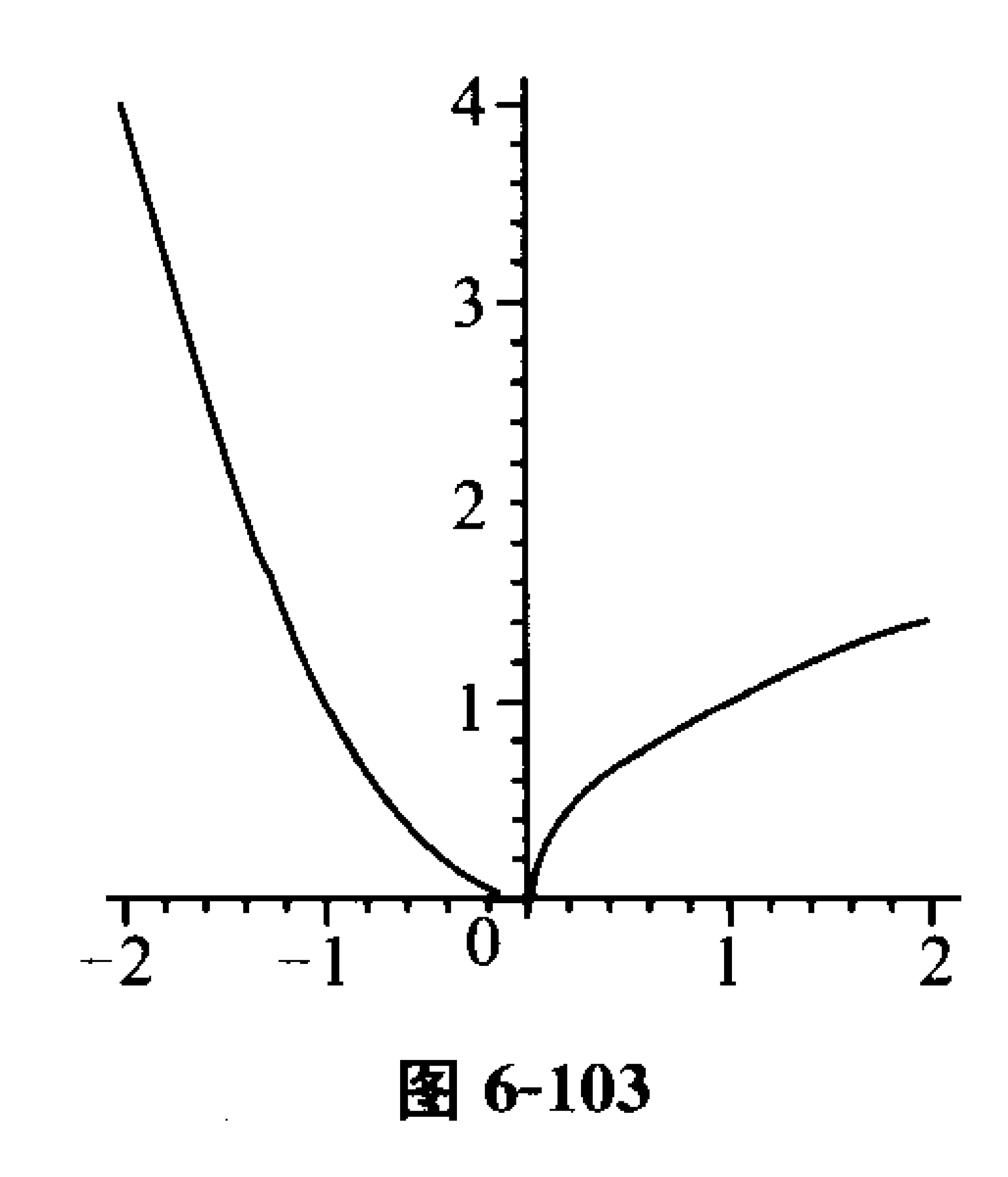


图 6-103

6.5 应用实例

【例 71】 考虑复平面上的迭代 $z_{k+1}=z_k^2+a$。绘制其 Julia 集 $J(a)=\{z_0:|z_k|<\infty,\forall_k\}$(图6-104) 与 Mandelbrot 集 $M(z_0)=\{a:|z_k|<\infty,\forall_k\}$(图 6-105)。

设定最大迭代次数 N,复数模长上界 M。对给定 z_0 或 a,若存在 $0\leqslant k\leqslant N$ 使得 $|z_k|>M$,我们则对点 z_0 或 a 用灰度等级 $n-k$ 来显示;否则用灰度等级 0 来显示。

```
> f := proc(a,z,m,n)
    local k,x; x:=evalf(z);
    for k from 0 to n-1 while abs(x)<=m do x:=x^2+a end do;
    return n-k
  end proc;
> Julia:=(x,y)->f(-1.25,x+y*I,100,20);
  plots[densityplot](Julia,-2..2,-2..2,grid=[100,100],
  axes=boxed,style=patchnogrid);
```

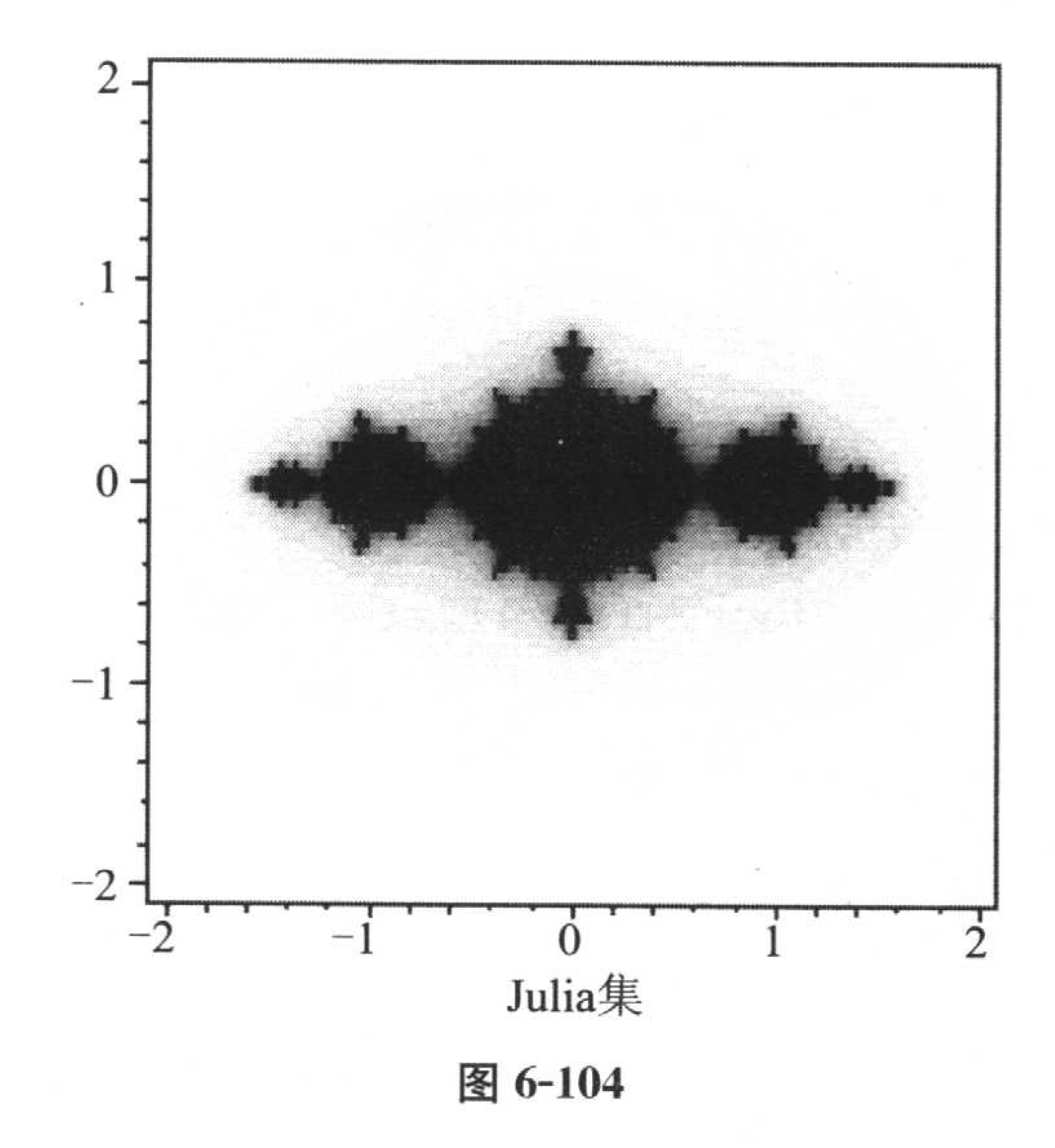


图 6-104

```
> Mandelbrot：=(x,y)->f(x+y*I,0.100,20);
  plots[densityplot](Mandelbrot,-2.5..1.5,-2..2,grid=[100,100],
  axes=boxed,style=patchnogrid);
```

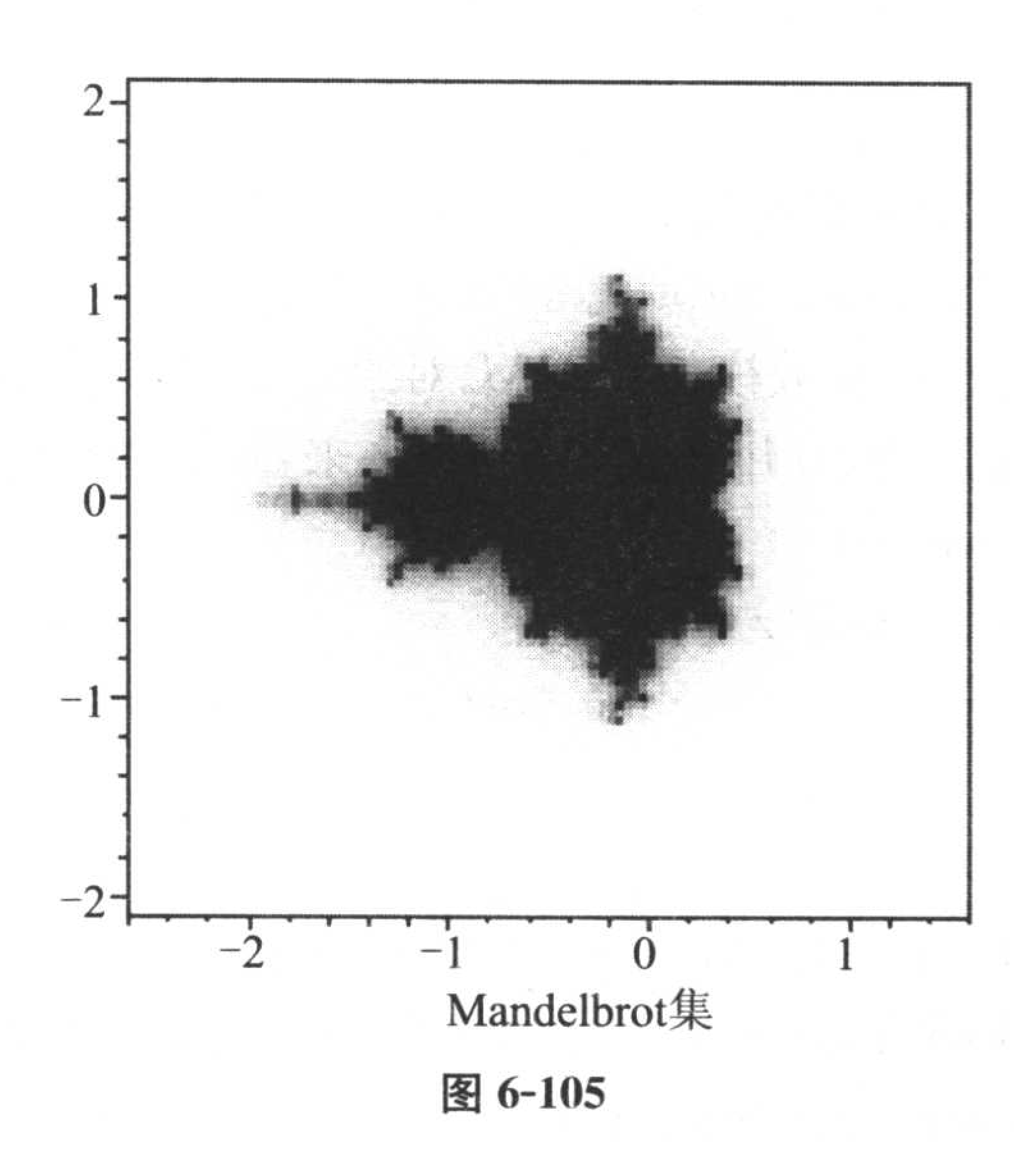


图 6-105

习　　题

1. 作出函数的图像。

(1) $y(x)=1+\sin x-x^2, x\in[-30,30]$

(2) $y(x)=(x+1)(x-2)(x+3)(x-4), x\in[-20,20]$

(3) $y(x)=\sin(x+\cos(x+\sin(x))), x\in[-10,10]$

2. 同时作出 $y(x)$和 $y'(x)$的图像。

(1) $y(x)=\dfrac{x^2(x-1)}{(x+1)^2}, x\in[-100,100]$

(2) $y(x)=\dfrac{\sin x}{1+x^2}, x\in[-90,90]$

3. 画出参数曲线。

(1) $x=\dfrac{(t+1)^2}{4}, y=\dfrac{(t-1)^2}{4}, t\in[-6,6]$

(2) $x=a\cos 2t, y=a\cos 3t, t\in[-\pi,\pi]$

(3) $x=t\ln t, y=\dfrac{\ln t}{t}, t\in[0,6\pi]$

(4) 圆的渐开线 $\begin{cases} x=\cos\theta+\theta\sin\theta \\ y=\sin\theta-\theta\cos\theta \end{cases}, 0\leqslant\theta\leqslant 4\pi$

4. 画出三维图形。

(1) $z=e^{-x^2-y^2}, -3\leqslant x, y\leqslant 3$

(2) $z=\dfrac{x^2-y^2}{x^3+y^3}, -10\leqslant x, y\leqslant 10$

(3) $z=\sin(x+\cos y), -6\leqslant x, y\leqslant 6$

5. 自取色彩,用极坐标画玫瑰线。

(1) 三瓣玫瑰线 $\rho=\sin 3\theta, 0\leqslant\theta\leqslant 4\pi$

(2) 八瓣玫瑰线 $\rho=\sin 4\theta, 0\leqslant\theta\leqslant 4\pi$

(3) $\rho=\sin\dfrac{4\theta}{3}, 0\leqslant\theta\leqslant 4\pi$

6. 自定 θ 范围,用极坐标画一朵花 $\rho=3\sin\theta+3.5\cos(10\theta)\cos\theta$。

7. 作出函数 $f(x,y)=\sin(x^2y)-\cos(y^2), -4\leqslant x, y\leqslant 4$ 的密度图和等值线图。

8. 自取 a,b,c,画出三维图形。

(1) 单叶双曲面 $\dfrac{x^2}{a^2}+\dfrac{y^2}{b^2}-\dfrac{z^2}{c^2}=1$

(2) 双叶双曲面 $\dfrac{x^2}{a^2}-\dfrac{y^2}{b^2}-\dfrac{z^2}{c^2}=1$

9. 画出悬链面 $\begin{cases} x=2\cosh\dfrac{t}{2}\cos\theta \\ y=2\cosh\dfrac{t}{2}\sin\theta \\ z=t \end{cases}, \begin{matrix} 0\leqslant t\leqslant 3 \\ 0\leqslant\theta\leqslant 4\pi \end{matrix}$。

10. 画出锥面 $\dfrac{x^2}{9}-\dfrac{y^2}{16}-\dfrac{z^2}{4}=1$。

11. 画出球面 $x^2+y^2+z^2=9$ 与锥面 $z=\sqrt{x^2+y^2}$ 所围立体及其表面。

12. 画出两球面 $x^2+y^2+z^2=9$ 与 $x^2+y^2+(z-3)^2=9$ 所围立体及其表面。

第 7 章　Maple 程序设计

7.1　流 程 控 制

在 Maple 中主要有两种类型的流程控制语句:条件语句和循环语句。

1. 条件语句(if 语句)

if 语句的一般形式为:

```
if 条件 1 then 语句序列 1
elif 条件 2 then 语句序列 2
elif 条件 3 then 语句序列 3
……
else 语句序列 4
end if
```

其中“elif 条件 then 语句序列”和“else 语句序列”均可以省略,而且“条件”必须为逻辑型表达式。if 还可以被用作为一个操作符,其格式为:

```
`if`(条件,表达式 1,表达式 2)
```

当“条件”取值为 true 时,返回“表达式 1”的值;当“条件”取值为 false 时,返回“表达式 2”的值。由于 if 为 Maple 的保留字,操作符 if 必须被夹在两个反引号当中来使用。Maple 11 中与 if 类似的保留字还有:and,done,implies,intersect,minus,mod,not,or,quit,stop,subset,union,xor 等。

【例 1】　分段连续函数 $f(x)=\begin{cases}0, & x\in(-\infty,0)\\ x^2, & x\in[0,1]\\ 2x-1, & x\in(1,+\infty)\end{cases}$。

```
> f :=proc(x)
```

```
    if x<0  then  0 elif x<1  then  x^2 else  2*x-1 end if
  end proc;
```

或者，更简单地，

```
> f:=x->`if(x<0,0,`if(x<1,x^2,2*x-1));
```

又或者，使用 piecewise 函数，

```
> f:=x-> piecewise(x<0,0,x<=1,x^2,2*x-1);
```

2. 循环语句(for..while..do 语句)

for..while..do 语句的一般形式为：

```
for 变量 from 初值 by 步长 to 终值 while 条件 do
   语句序列
end do
```

或者

```
for 变量 in 列表 while 条件 do
   语句序列
end do
```

其中“for 变量”，“from 初值”，“by 步长”，“to 终值”，“while 条件”和“in 列表”各部分均可以省略，“初值”和“步长”的缺省值都是 1，“终值”的缺省值是 infinity，“条件”的缺省值是 true，而且“条件”必须为逻辑型表达式。

【例 2】　今有物不知其数，三三数之剩二，五五数之剩三，七七数之剩二，问物几何？

```
> for x from 2 by 7 do
     if x mod 3=2 and  x mod 5=3 then print(x); break end if
  end do;
```

23

本例中，当第一个满足条件的正整数被发现后，我们使用了 break 语句来退出循环体。否则，程序将无限循环下去，输出所有解。如果我们希望跳过本次循环中的剩下语句而继续下一轮循环的话，则可以使用 next 语句。例如：

```
> for x do
     if x=3  then next end if;
     if x=6 then break end if;
     print(x)
```

```
end do;
```

$$1$$
$$2$$
$$4$$
$$5$$

【例 3】 用 Newton 迭代法 $x_{k+1}=x_k-\dfrac{f(x_k)}{f'(x_k)}$ 求方程 $f(x)=e^x-x-2=0$ 在 $x_0=1$ 附近的根，直到 $|x_k-x_{k+1}|<10^{-9}$。

```
> x := 2.0: y := 1.0:
  while abs(y)>=1e-9  do
    x := x-y; y := (exp(x)-x-2)/(exp(x)-1)
  end do:
  x;
```

$$1.146193221$$

【例 4】 求数据表的均值和方差。

```
> f := proc(a)
    local s1,s2,x;
    n :=nops(a); s1 :=0; s2 :=0;
    for x in a do s1 :=s1+x; s2 :=s2+x^2 end do;
    if n>0 then
       printf("mean = %f, variance = %f",s1/n,sqrt(s2-s1^2/n))
    end if
  end proc:
> f([1,2,3,4]);
```

mean = 2.500000, variance = 2.236068

【例 5】 显示连分式。

```
> x :=a[5]: for i from 5 by -1 to 1 do x := a[i-1]+1/x end do: x;
```

$$a_0+\cfrac{1}{a_1+\cfrac{1}{a_2+\cfrac{1}{a_3+\cfrac{1}{a_4+\cfrac{1}{a_5}}}}}$$

【例 6】 用二重循环语句输出 n 阶循环矩阵 $A=\begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ x_2 & \iddots & \iddots & x_1 \\ \vdots & \iddots & \iddots & \vdots \\ x_n & x_1 & \cdots & x_{n-1} \end{pmatrix}$。

```
> n:=5; A:=Matrix(n); for i to n do for j to n do
  A[i,j]:=x[modp(i+j-2,n)+1] end do end do; A;
```

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 \\ x_2 & x_3 & x_4 & x_5 & x_1 \\ x_3 & x_4 & x_5 & x_1 & x_2 \\ x_4 & x_5 & x_1 & x_2 & x_3 \\ x_5 & x_1 & x_2 & x_3 & x_4 \end{bmatrix}$$

除了条件语句和循环语句之外，Maple 还提供了一些具有特殊用途的命令。这些预定义的命令大大减轻了我们的编程工作量，提高了程序的运行效果和可读性。虽然这些命令中的一部分已经在前面各章中有所叙述，但是出于完整性我们仍然将它们罗列于此。

3. $ 操作符

$ 操作符的用途是通过替换"表达式"中"变元"的值来构造一个序列。其用法有：

> 表达式 $ 变元 = 初值..终值
> 表达式 $ 重复次数
> $ 初值..终值

例如：

```
> x^2 $ x=1..3;
```

1, 4, 9

```
> x, $ 3;
```

x, x, x

```
> $ 1..3;
```

1, 2, 3

4. seq 语句

seq 语句的用途也是通过替换"表达式"中"变元"的值来构造一个序列。其用法有：

```
seq(表达式,变元 = 初值..终值,步长)
seq(表达式,变元 = 指标集)
seq(表达式,变元 in 指标集)
seq(初值..终值,步长)
```

其中“步长”的缺省值是 1,可省略,指标集通常为一个数组、列表或集合。例如:

```
> seq(x^2, x=3..7, 2);
                                9, 25, 49
> seq(x^2, x={3,5,7});
                                9, 25, 49
> seq(x^2, x in [3,5,7]);
                                9, 25, 49
> seq(3..7, 2);
                                3, 5, 7
```

5. select 语句,remove 语句和 selectremove 语句

select 语句的用途是从“表达式”中选出使“过程”为真值的运算量;remove 语句的用途是从“表达式”中选出使“过程”为假值的运算量;selectremove 语句的用途是将“表达式”的运算量分成两部分,一部分使“过程”为真值,另一部分使“过程”为假值。其用法为:

```
select(过程,表达式,参数序列)
remove(过程,表达式,参数序列)
selectremove(过程,表达式,参数序列)
```

其中“过程”的返回值必须是逻辑值,“参数序列”为“过程”的第 2,3,… 个参数。例如:

```
> ismultiple := proc(x,y:=2) mod(x,y)=0 end proc;
> select(ismultiple, [6,5,4,3,2,1], 3);
                                6, 3
> remove(ismultiple, [6,5,4,3,2,1], 3);
                                5, 4, 2, 1
> selectremove(ismultiple, {6,5,4,3,2,1});
                                {2,4,6}, {1,3,5}
> select(type, {ismultiple,sin(x)}, procedure);
                                [ismultiple]
```

```
> remove(has, x^2 + x y + y^2, y);
```

$$x^2$$

6. map 语句

map 语句的用途是将“过程”分别作用于“表达式”的各个运算量。其用法有：

```
map(过程,表达式,参数序列)
map2(过程,参数 1,表达式,参数序列)
map[n](过程,参数 1,…,参数 n-1,表达式,参数序列)
map[evalhf](过程,表达式,参数序列)
map[inplace](过程,表达式,参数序列)
```

当选项 n 为正整数时，“参数 1”，…，“参数 $n-1$”为“过程”的前 $n-1$ 个参数，“参数序列”为“过程”的第 $n+1,n+2,\cdots$ 个参数。当“表达式”的类型为 rtable 时，选项 evalhf 使得“过程”数值地作用于“表达式”的各个运算量，其效果相当于：

evalhf(map(过程,表达式,参数序列))；

选项 inplace 使得“表达式”的值同时被更新，其效果相当于：

表达式 := map(过程,表达式,参数序列)。

当“表达式”的类型不是 rtable 时，选项 evalhf 和 inplace 则不起作用。例如：

```
> A := rtable(1..2,1..2,[[1,2],[3,4]]);
```

$$A := \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

```
> map(sqrt,A);
```

$$\begin{bmatrix} 1 & \sqrt{2} \\ \sqrt{3} & 2 \end{bmatrix}$$

```
> unassign(f); map[3](f,x,y,A,z);
```

$$\begin{bmatrix} f(x,y,1,z) & f(x,y,2,z) \\ f(x,y,3,z) & f(x,y,4,z) \end{bmatrix}$$

```
> map[evalhf](sqrt,A);
```

$$\begin{bmatrix} 1. & 1.41421356237309515 \\ 1.73205080756887720 & 2. \end{bmatrix}$$

```
> map[inplace](f,A,x);
```

$$\begin{bmatrix} f(1,x) & f(2,x) \\ f(3,x) & f(4,x) \end{bmatrix}$$

```
> A;
```

$$\begin{bmatrix} f(1,x) & f(2,x) \\ f(3,x) & f(4,x) \end{bmatrix}$$

7. zip 语句

zip 语句的用途是将"过程"作用于"列表 1"和"列表 2"的相应元素，由此产生一个新的列表。其用法为：

zip(过程，列表 1，列表 2，缺省值)

其中"列表 1"和"列表 2"也可以是数组、向量或者矩阵，但不可以是集合；"缺省值"被用来添加在列表后面使得两个列表有着同样的大小；当"缺省值"被省略时，zip 语句产生的列表的大小为"列表 1"和"列表 2"大小的最小值。例如：

```
> a := matrix(2,3,[[1,2,3],[4,5,6]]);
```

$$a := \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

```
> b := matrix(3,2,[[u,v],[x,y],[z,w]]);
```

$$b := \begin{bmatrix} u & v \\ x & y \\ z & w \end{bmatrix}$$

```
> zip((x,y)-> x+y,a,b);
```

$$\begin{bmatrix} 1+u & 2+v \\ 4+x & 5+y \end{bmatrix}$$

```
> zip((x,y)-> x+y,a,b,NAN);
```

$$\begin{bmatrix} 1+u & 2+v & 3+NAN \\ 4+x & 5+y & 6+NAN \\ NAN+z & NAN+w & 2NAN \end{bmatrix}$$

8. add 语句和 mul 语句

add 语句的用途是对一个序列数值求和；mul 语句的用途是对一个序列数值乘积。它们的用法有：

add(表达式，变元 = 初值..终值)
add(表达式，变元 = 指标集)
add(表达式，变元 in 指标集)
mul(表达式，变元 = 初值..终值)
mul(表达式，变元 = 指标集)
mul(表达式，变元 in 指标集)

例如：

```
> add(x^3, x=1..3);
```

$$36$$

```
> add(x^3, x={1,2,3});
```

$$36$$

```
> add(x^3, x in {1,2,3});
```

$$36$$

9. sum 语句和 product 语句

sum 语句的用途是对一个序列符号求和；product 语句的用途是对一个序列符号乘积。它们的用法有：

```
sum(表达式,变元)
sum(表达式,变元 = 初值..终值)
sum(表达式,变元 = 多项式的根集)
sum(表达式,变元 = 表达式)
product(表达式,变元)
product(表达式,变元 = 初值..终值)
product(表达式,变元 = 多项式的根集)
product(表达式,变元 = 表达式)
```

例如：

```
> sum(x^n, n);
```

$$\frac{x^n}{x-1}$$

sum(f(x),x)的返回值为某个 $g(x)$ 满足对任意 x 都有 $g(x+1)-g(x)=f(x)$。

```
> sum(x^(2n+1)/(2n+1)!, n=0..infinity);
```

$$\sinh(x)$$

```
> sum(x^n, n=10..5);
```

$$-x^6-x^7-x^8-x^9$$

当 $m\leqslant n$ 时，sum(f(x),x=m..n)的返回值为 $f(m)+\cdots+f(n)$；当 $m=n+1$ 时，sum(f(x),x=m..n)的返回值为 0；当 $m\geqslant n+2$ 时，sum(f(x),x=m..n)的返回值为 $-(f(n+1)+\cdots+f(m-1))$。

```
> sum(x, x=RootOf(x^2+2x+1));
```

$$-2$$

sum(f(x),x=RootOf(g))的返回值为 $\sum\limits_{g(x)=0} f(x)$ 。

```
> sum(x^n, n=2);
```

$$x^2$$

sum(f(x),x=表达式)的返回值为 f(表达式)。

10. Sum 语句和 Product 语句

Sum 语句和 Product 语句分别是 sum 语句和 product 语句的惰性形式,用法同上。

7.2 过程(Procedure)

Maple 过程由一系列 Maple 语句组成,相当于各种编程语言中的子程序。一个完整的 Maple 过程通常具有以下形式:

```
过程名 := proc(参数序列)::返回值类型
    local 局部变量序列;
    global 全局变量序列;
    uses 需用的函数包序列;
    options 选项序列;
    description 对过程的描述和说明;
    具体的语句;
end proc
```

若一个变量没有明确声明为全局变量,则为局部变量。过程的返回值为所执行的最后一条语句的返回值。

【例 7】 Hello,World!

```
> p:=proc() printf("Hello, World!") end proc;
```

为了增加程序的可读性,我们可以用对齐(插入空格)和分行(Shift + Enter)来美化程序。

```
> p:=proc()
      printf("Hello, World!")
  end proc;
```

上述语句的结果均为 Maple 过程 p 的定义语句

p := proc() printf("Hello, World!") end proc

我们可以使用 print 语句或 eval 语句来查看过程的定义语句。运行过程 p 时,

键入

> p()；

得到结果

Hello, World!

若我们省略了 p 后面的()，则结果为

> p;

$$p$$

【例 8】 计算一个向量 v 的 L_p 范数 $\|v\|_p=\left(\sum|v_i|^p\right)^{1/p}$。

```
> pnorm := proc(v,p:=2)
      local n,s,x;
      description "Compute the L_p norm of a vector";
      n:=nops(v); if n=0 then return 0 end if;
      if p=0 then   # return the geometric mean
          s:=1; for x in v do s:=s*abs(x) end do; return s^(1/n)
      elif p=infinity then
          s:=0; for x in v do s:=max(s,abs(x)) end do; return s
      else
          s:=0; for x in v do s:=s+abs(x)^p end do; return s^(1/p)
      end if
  end proc;
```

使用该过程时，例如：

> pnorm([1,2,3,4],1);

$$10$$

> a:=rtable(1..2,1..2,[[1,2],[3,4]]); pnorm(a);

$$a:=\begin{bmatrix}1 & 2\\ 3 & 4\end{bmatrix}$$

$$\sqrt{30}$$

程序的第 1 行定义了过程的名称 pnorm，输入参数 v 和 p，并设 p 的缺省值为 2。为了程序的通用性，我们没有明确输入参数的类型。如果我们限制 v 为数值型列表，p 为整数，则这一行可重写为

pnorm := proc(v::list(numeric),p::integer:=2)。

程序的第 2 行定义了过程所用到的 3 个局部变量 n,s,x。程序的第 3 行为过程的描述语句，可以通过 Describe 语句来查看。

> Describe(pnorm);

```
# Compute the L_p norm of a vector
pnorm(v, p := 2)
```

程序的第 4～11 行为过程的具体执行语句。最后，过程以 end proc 结束。在程序的第 5 行中，# 及其后面的部分为注释，对程序不起任何作用。

程序的第 4,6,8,10 行中，我们分别使用了 return 语句来提前结束程序的运行并返回计算结果。

【例 9】 建立 n 以内的素数表。

```
> PrimeTable := proc(n::integer)
    local i,j,k,m,p;
    description "List all primes no more than n";
    if n<2 then return NULL end if;
    p := 2;
    for m from 3 by 2 to n do
      j := 1; k := trunc(sqrt(m));
      for i in p do
        if i>k then break elif m mod i=0 then j := 0; break end if
      end do;
      if j=1 then p := p,m end if
    end do;
    p
  end proc;
```

在编写 Maple 过程时，需要注意的是：

- 输入参数可以为空，但 proc 后的()不可省。
- 输入参数为只读类型，通过值传递，不可被修改。
- 若欲修改过程外部变量的值，则必须在过程内声明此变量为全局变量。
- 尽管任何未经声明的变量都默认为局部变量，最好还是明确声明。
- 局部变量与过程外的其他任何同名变量无关。
- 避免使用未赋初值的局部变量。
- 除了注释之外，不要使用中文字符(以及标点符号、文件名、目录等)。

例如：

```
> p := proc() a := 1; a+b end proc;
Warning, `a` is implicitly declared local to procedure `p`
```

$$p := \text{proc}()\ \text{local}\ a;\ a := 1;\ a+b\ \text{end proc}$$

在过程 p 中，a 被认为是一个局部变量，但 b 却被认为是一个符号而不是一个

变量。

```
> b := 1; p();
```

$$b := 1$$

$$2$$

```
> b := 2; p();
```

$$b := 2$$

$$3$$

【例 10】 求任意个多项式的最大公因式。

```
> pgcd := proc()
      local i,x;
      if nargs=0 then return end if;
      x :=args[1];
      for i from 2 to nargs do
         x :=gcd(x,args[i])
      end do;
      x
  end proc;
```

在 Maple 中已经有了计算两个多项式的最大公因式的函数 gcd(a,b)，我们只需将其推广到任意个参数形式。由于参数的个数不确定，过程没有设定参数，但是我们可以在程序中通过特殊变量 args 和 nargs 来获取过程的参数列表和过程的参数个数。

除了 args 和 nargs 之外，在 Maple 过程中我们还可以使用特殊变量 procname，_passed，_params，_options，_rest，_npassed，_nparams，_noptions，_nrest，_nresults，其中 procname 为过程的名称。例如：

```
> f := proc() [args, nargs, procname] end proc;
> f((x+1)^2-x^2-2*x,`mod`,(1+x)-(1-x));
```

$$[(x+1)^2-x^2-2x, modp, 2x, 3, f]$$

关于特殊变量的具体用法，请参阅 Maple 的联机帮助、用户手册以及相关的书籍资料。除了以上标准的过程定义语句之外，Maple 还提供一些简便的方法。

1. -> 操作符

Maple 中的 -> 操作符是一种特殊的过程，它具有形式：

变量列表 -> 结果

并且等价于语句

proc(变量列表) option operator,arrow; 结果 end　proc

例如：

> f := x -> (1,x,x^2);

$$f := x \to 1, x, x^2$$

注意：由于"->"的优先级要高于","的优先级，1，x，x^2 一定要在()中。

> g := (a,b) -> proc(x) sin(a * x+b) end proc();

$$g := (a,b) \text{->} \text{proc}(x)\ \sin(ax+b)\ \text{end proc}$$

> plot(g(1,1));

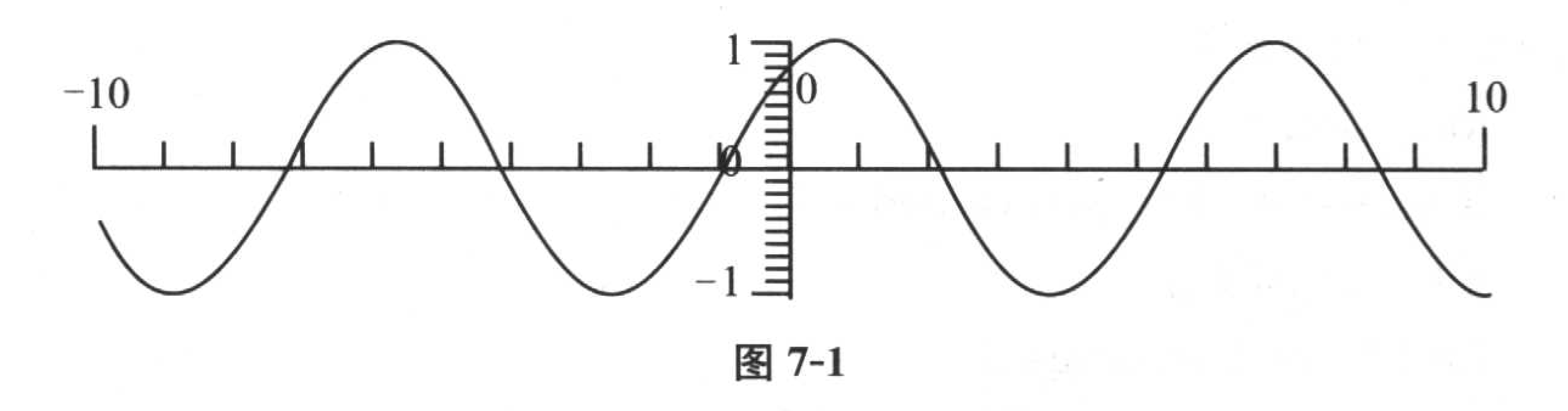


图 7-1

2. unapply 语句

当表达式不是一个列表的时候，

unapply(表达式，变量列表，选项列表)

将表达式转化为一个过程

变量列表 -> 表达式

例如：

> f := unapply(x^2,x);

$$f := x \text{->} x\hat{}2$$

> g := unapply([f(x),f(y)],x,y);

$$g := (x,y) \text{->} [x\hat{}2, y\hat{}2]$$

又如：

> a :=1; f := unapply(x+a,x); g := x -> x+a;

$$a := 1$$

$$f := x \text{->} x+1$$

$$g := x \text{->} x+a$$

> a :=2; f(x); g(x);

$$a := 2$$

$$x+1$$

$$x+2$$

-> 操作符和 unapply 语句的效果并不完全相同。

【例 11】　给定任意多个点 (x_i, y_i)，构造 Lagrange 插值多项式 $f(t) = \sum_i \left(y_i \prod_{j \neq i} \frac{t - x_j}{x_i - x_j}\right)$。

```
> interpolation := proc(p)
    local g,i,j,n,x;
    n:=nops(p);
    if n<2 then error "Insufficient points!" end if;
    try
      g:=add (mul((x-p[j,1])/(p[i,1]-p[j,1]), j=1..i-1)
                * mul((x-p[j,1])/(p[i,1]-p[j,1]), j=i+1..n)
                * p[i,2], i=1..n);
      return unapply(sort(simplify(g)),x)
    catch:
      print("The x-coordinate of points must be distinct.");
      return unknown
    end try
  end proc;
```

在程序的第 4 行中，error 语句的作用是显示错误信息并立即中断程序的运行。程序的第 9 行返回一个过程。注意：此处不可以用语句

```
x -> sort(simplify(g))
```

来代替。第 5 行到第 13 行的 try…catch 语句块被用来处理可能出现的蜕化情形。详见 7.6 节。

```
> f := interpolation({[1,1],[2,3],[3,5],[4,3],[5,1]});
```

$$f := x \rightarrow \frac{1}{3}x^4 - 4x^3 + \frac{47}{3}x^2 - 22x + 11$$

```
> g := interpolation({[1,1],[3,2],[5,3],[3,4],[1,5]});
              "The x-coordinate of points must be distinct."
                          g :=unknown
> plot(f(t),t=1..5);
```

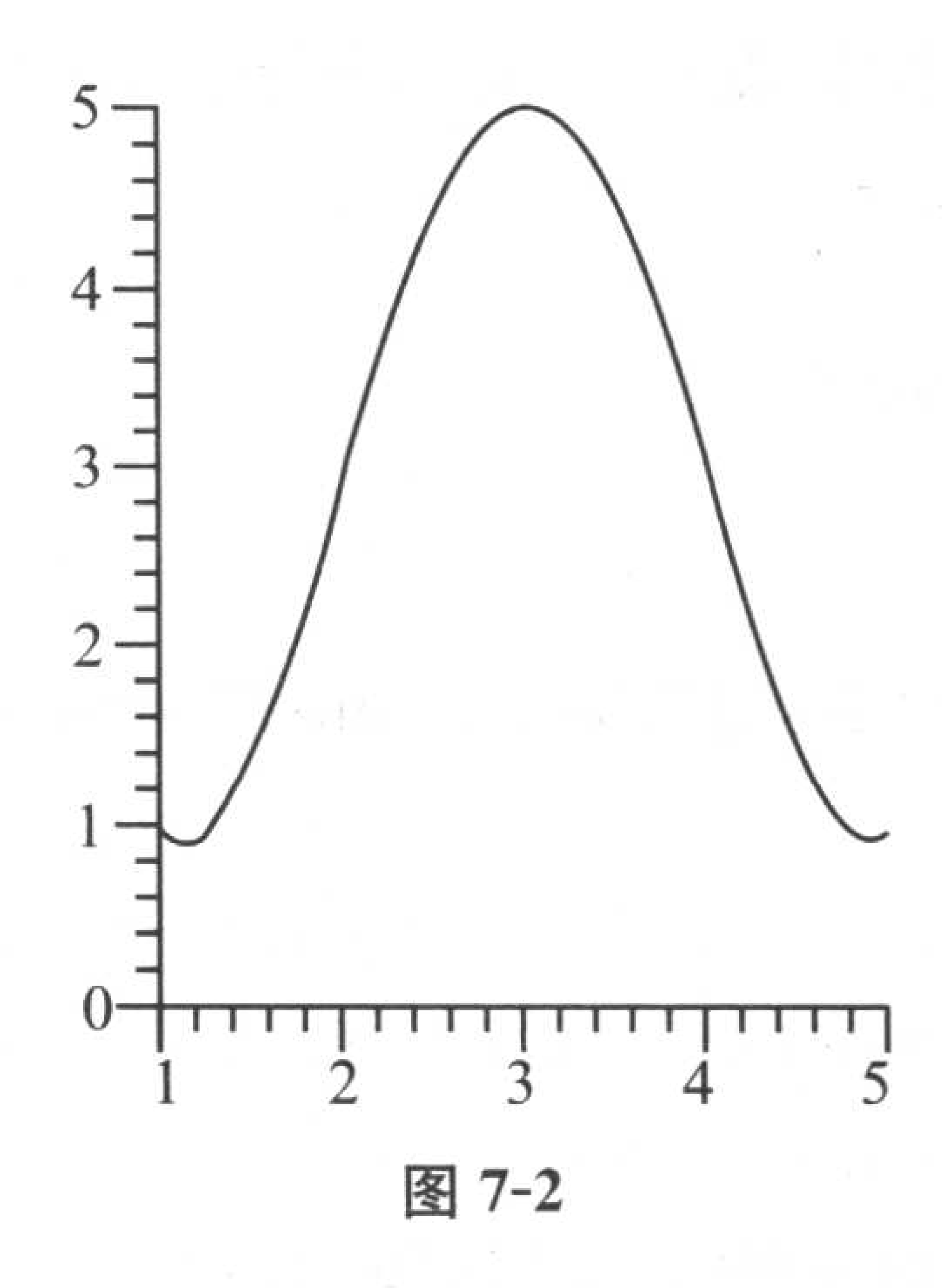


图 7-2

7.3　模块(Module)

Maple 模块是最高级的 Maple 表达式，相当于面向对象的编程语言中的类(Class)。Maple 模块的定义与 Maple 过程的定义非常相似，都是由一系列 Maple 语句组成，通常具有以下形式：

```
模块名 := module()
    export 导出变量序列;
    local 局部变量序列;
    global 全局变量序列;
    option 选项序列;
    description 对模块的描述和说明;
    具体的语句;
end module
```

简单起见，我们可以认为 Maple 模块提供了一批预先定义好了的变量，这些被导出(通过模块中的 export 语句)的变量可以被模块外部的 Maple 语句所使用，它们可以是符号、数据、对象，也可以是 Maple 过程或 Maple 模块，甚至是该模块自己。当模块被生成的时候，模块中的语句被依次执行。

【例 12】　一个二阶方阵的模块。

> M := module()

```
export Print,SetData,Det,Inverse,Rank,Trace,Transpose;
description "A sample module of 2-by-2 matrix";
local a; a:=array([[0,0],[0,0]]); # matrix storage
Print:= proc()
    description "Print the matrix";
    print(a)
end proc;
SetData:= proc(b::list)
    description "Set values for the matrix";
    a:=array(1..2,1..2,b)
end proc;
Det:= proc()
    description "Determinant";
    a[1,1]*a[2,2]-a[1,2]*a[2,1]
end proc;
Inverse:= proc()
    description "Inverse matrix";
    local d; d:=a[1,1]*a[2,2]-a[1,2]*a[2,1];
    if d=0 then error "Matrix is singular!" end if;
    array([[a[2,2]/d,-a[1,2]/d],[-a[2,1]/d,a[2,2]/d]])
end proc;
Rank:= proc()
    description "Matrix rank";
    if a[1,1]*a[2,2]-a[1,2]*a[2,1]<>0 then 2
    elif a[1,1]=0 and a[1,2]=0 and a[2,1]=0 and a[2,2]=0 then 0
    else 1 end if
end proc;
Trace:= proc()
    description "Sum of the diagonal elements";
    a[1,1]+a[2,2]
end proc;
Transpose:= proc()
    description "Transpose matrix";
    array([[a[1,1],a[2,1]],[a[1,2],a[2,2]]])
end proc;
```

end module;

与 Maple 过程的定义语句不同，上述语句的输出结果为：

M：= module() description "A sample module of 2-by-2 matrix"; local *a*; export Print, SetData, Det, Inverse, Rank, Trace, Transpose; end module

其中不含有任何具体的执行语句，而只是对模块结构的描述。此模块有一个局部变量 *a* 用来储存矩阵的元素，并且提供了 7 个导出变量 Print, SetData, Det, Inverse, Rank, Trace, Transpose 让用户使用。用户可通过 Describe 语句或 exports 语句来查看模块所导出的变量名称，并且通过：- 操作符来引用模块的导出变量。

exports(模块,选项)

模块：- 变量

```
> Describe(M);
# A sample module of 2-by-2 matrix
module M:
    # Print the matrix
    Print()
    # Set values for the matrix
    SetData(b::list)
    # Determinant
    Det()
    # Inverse matrix
    Inverse()
    # Matrix rank
    Rank()
    # Sum of the diagonal elements
    Trace()
    # Transpose matrix
    Transpose()
> exports(M);
```

Print, SetData, Det, Inverse, Rank, Trace, Transpose

使用模块 *M* 的时候，首先，我们对矩阵赋值

```
> M:-SetData([[1,2],[3,6]]); M:-Print();
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 6 \end{bmatrix}$$

然后，我们可以计算矩阵的行列式、逆矩阵、秩、迹、转置矩阵。

```
> M:-Det();
                                  0
> M:-Inverse();
                Error, (in Inverse) Matrix is singular!
> M:-Rank();
                                  1
> M:-Trace();
                                  7
> M:-Transpose();
```

$$\begin{bmatrix} 1 & 3 \\ 2 & 6 \end{bmatrix}$$

在某些情况下，我们可以在定义模块的同时给模块起一个名字，这种模块称为具名模块(named module)。例如：我们可以将例 10 的第一行改为

```
> module MATRIX()
```

这时，M：—与 MATRIX：—具有相同的效果。它们的差别在于：具名模块的变量名 MATRIX 受到保护，从而无法被赋值，而普通模块的变量名可被反复重新赋值。

```
> module m1() export e1; end module;
                module m1() export e1; end module
> m1 := module () export e2; end module;
          Error, attempting to assign to `m1` which is protected
```

与 constructor 和 destructor 相对应，Maple 模块中的特殊变量 ModuleLoad 和 ModuleUnload 起着类似的作用。模块中的特殊变量还有 ModulePrint，Module Apply和 thismodule。ModuleLoad，ModuleUnload，ModulePrint 可以被定义为局部变量或导出变量，ModuleApply 必须被定义为导出变量，thismodule 则代表该模块自身。例如：

```
> m := module()
    export e,ModuleApply;
    local ModuleLoad,ModuleUnload;
    e := thismodule;
    ModuleApply := proc() max(args) end proc;
    ModuleLoad := proc() print("Load Module\n") end proc;
    ModuleUnload := proc() print("Unload Module\n") end proc;
  end module;
  m:= module() local ModuleLoad, ModuleUnload; export e, ModuleApply;
  end module
```

```
> m:-e(1,x,3)
```

$$\max(3,x)$$

7.4 函数包(Package)

Maple 函数包是一些程序的集合。这些程序通常是为了解决某类问题而被放在一起的。例如：Maple 11 就提供了包括 algcurves，combinat，CurveFitting，DETools，finance，geom3d，geometry，group，IntegrationTools，LinearAlgebra，Logic，Maplets，MatrixPolynomialAlgebra，numapprox，numtheory，Optimization，PDETools，plots，PolynomialTools，RandomTools，RootFinding，SNAP，SolveTools，Statistics，Student，tensor，VectorCalculus 等在内的约 100 个函数包给用户使用。除了一些陈旧的函数包之外，绝大多数的函数包都是以模块的形式定义。因此，我们可以简单地认为一个函数包就是一个特殊的模块。我们只需在模块的定义语句里面包含 option package；或者将一个现有的模块通过 convert 语句转化为函数包。然后通过 savelib 语句将函数包存入一个程序库中。

```
convert(模块,package)
savelib(变量名)
```

这个程序库位于系统全局变量 savelibname 或 libname 所包含的路径中，通常是一个 *.lib 文件和一个 *.ind 文件(maple.lib，maple.ind)或是一个单一的 *.mla 文件(maple.mla)。在不同版本的 Maple 中，库文件的格式也不尽相同。

【例 13】 一个关于二阶方阵的函数包。

```
> module M()
    export Print,SetData,Det,Inverse,Rank,Trace,Transpose;
    local a;
    option package;
    description "A sample package of 2-by-2 matrix";
    ……   # 语句同例 6
  end module;
> savelib('M');
Error, cannot open archive, C:\Program Files\Maple 11/lib, for writing.
```

当函数包被成功地写入程序库，savelib 语句的返回值为空(NULL)；否则，我们将会得到错误信息。本例中，我们没有可供写入的程序库，于是我们需要新建一个程

序库，在更新了变量 savelibname 或 libname 之后，再将函数包存入程序库中。

```
> march('create',"c:\\Program Files\\Maple 11\\lib\\myMaple.mla");
  libname := libname,"c:\\Program Files\\Maple 11\\lib\\myMaple.mla";
  savelib('M');
```

march 是 Maple 中用来管理库文件的语句，其一般形式为：

march(操作，文件路径，选项)

从此以后，只要程序库位于 libname 所包含的路径中，我们都可以不经定义模块 M 而直接使用 M 的导出变量。其效果就如同已经执行了模块 M 的定义语句一般。此处 M 被定义为一个具名模块，当然也可以被定义为一个普通模块。

有的时候，我们希望省去“模块名：-变量名”中“模块名：-”，而直接引用“变量名”。with 语句则提供了这样一种方便，unwith 语句则可以撤销 with 语句，恢复到正常状态。

with(函数包，变量 1，变量 2，…)
unwith(函数包)

例如：

```
> with(M);
        [Det, Inverse, Print, Rank, SetData, Trace, Transpose]
> SetData([[1,2],[3,4]]);
```

$$\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$$

```
> Det();
```

$$-2$$

```
> unwith(M);
> Det();
                    Det()
```

with 语句和 unwith 语句只能在 Maple 的用户界面下使用，而不可以出现在过程或模块中。若我们希望在过程或模块中省去“模块名：-”，则可以使用 uses 语句列出所有需用的函数包，或者使用 use 语句。

7.5　应用程序(Maplet)

Maplet 是一个图形化的 Maple 应用程序，它可以拥有窗口、菜单、按钮、工具栏、

文本框、对话框等组件，可以接受用户的鼠标和键盘输入，并且可以将计算结果以图形化的方式输出。Maplet 本质上是一个 JavaApplet。在 Maple 中有两种方法可以生成一个 Maplet。

1. 使用 Maple 的 Maplets 函数包生成 Maplet

【例 14】 Hello,World!

```
> use Maplets :-Elements in
      h := Maplet(["Hello,World!",Button("OK",Shutdown())])
  end use;
  Maplets :-Display(h)
```

h := Maplets :-Elements :-com. maplesoft. maplets. elements. Mmaplet
("locale"="Chinese_People's Republic of China. 936",
"onstartup"="_Maplets_reference_72",
Maplets :-Elements :-com. maplesoft. maplets. elements. MAction
("reference"="_Maplets_reference_67", "run"="false",
Maplets :-Elements :-com. maplesoft. maplets. elements. MCommandShutdown()),
Maplets :-Elements :-com. maplesoft. maplets. elements. MWindow
("reference"="_Maplets_reference_68",
"layout"="_Maplets_reference_69", "visible"="true",
"title"="Maplet", "resizable"="true", "enabled"="true"),
Maplets :-Elements :-com. maplesoft. maplets. elements. MBoxLayout
("reference"="_Maplets_reference_69", "visible" = "true",
"border"="false", "vertical"="false", "inset"="5",
Maplets :-Elements :-com. maplesoft. maplets. elements. MBoxColumn
("visible"="true", "border"="false", "spacing"="5",
"hscroll"="never", "vscroll"="never", "inset"="5",
Maplets :-Elements :-com. maplcsoft. maplets. elements. MBoxCell
("value"="_Maplets_reference_70", "hscroll"="never",
"valign"="center", "fill"="none", "vscroll"= "never",
"halign"="center"),
Maplets :-Elements :-com. maplesoft. maplets. elements. MBoxCell
("value"="_Maplets_reference_71", "hscroll"="never",
"valign"="center", "fill"="none", "vscroll"="never",
"halign"="center"))),

```
Maplets:-Elements:-com.maplesoft.maplets.elements.MLabel
("reference"="_Maplets_reference_70", "caption"="Hello, World!",
"visible"="true", "valign"="center", "halign"="center",
"enabled"="true"),
Maplets:-Elements:-com.maplesoft.maplets.elements.Mbutton
("reference"="_Maplets_reference_71", "caption"="OK",
"onclick"="_Maplets_reference_67", "visible"="true",
"valign"="center", "halign"="center", "enabled"="true"),
Maplets:-Elements:-com.maplesoft.maplets.elements.MAction
("reference"="_Maplets_reference_72", "run"="false",
`Maplets:-Elements:-com.maplesoft.maplets.elements.
MCommandRunWindow`("window" = "_Maplets_reference_68")),
module() export References, Expressions, Procedures,
NumberOfVariables, SetGetLookUp; end module)
```

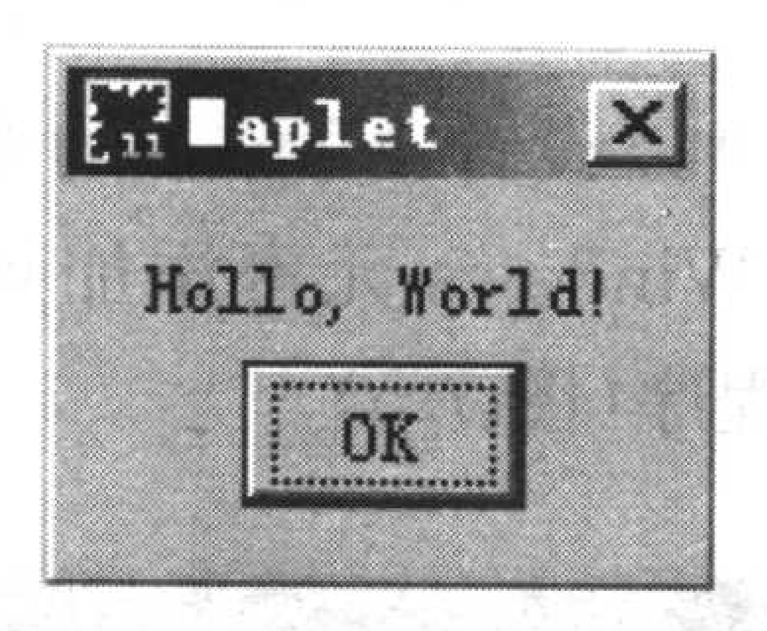


图 7-3

use 语句使我们省去了长长的“模块名：-”，程序也看起来十分简洁。其一般用法为：

```
use 局部变量约束 in
    语句序列
end use
```

例如：

```
> a:=1; use a=2 in a end use; a;
                         2
                         1
> use `+` = ((a,b)->a*b) in 3 + 4 end use; 3 + 4;
                         12
                         7
```

2. 利用开发工具 Maplet Builder 生成 Maplet

我们可以在 Maple 的菜单项“工具→分析助手”中找到 Maplet 生成器(图 7-4)。

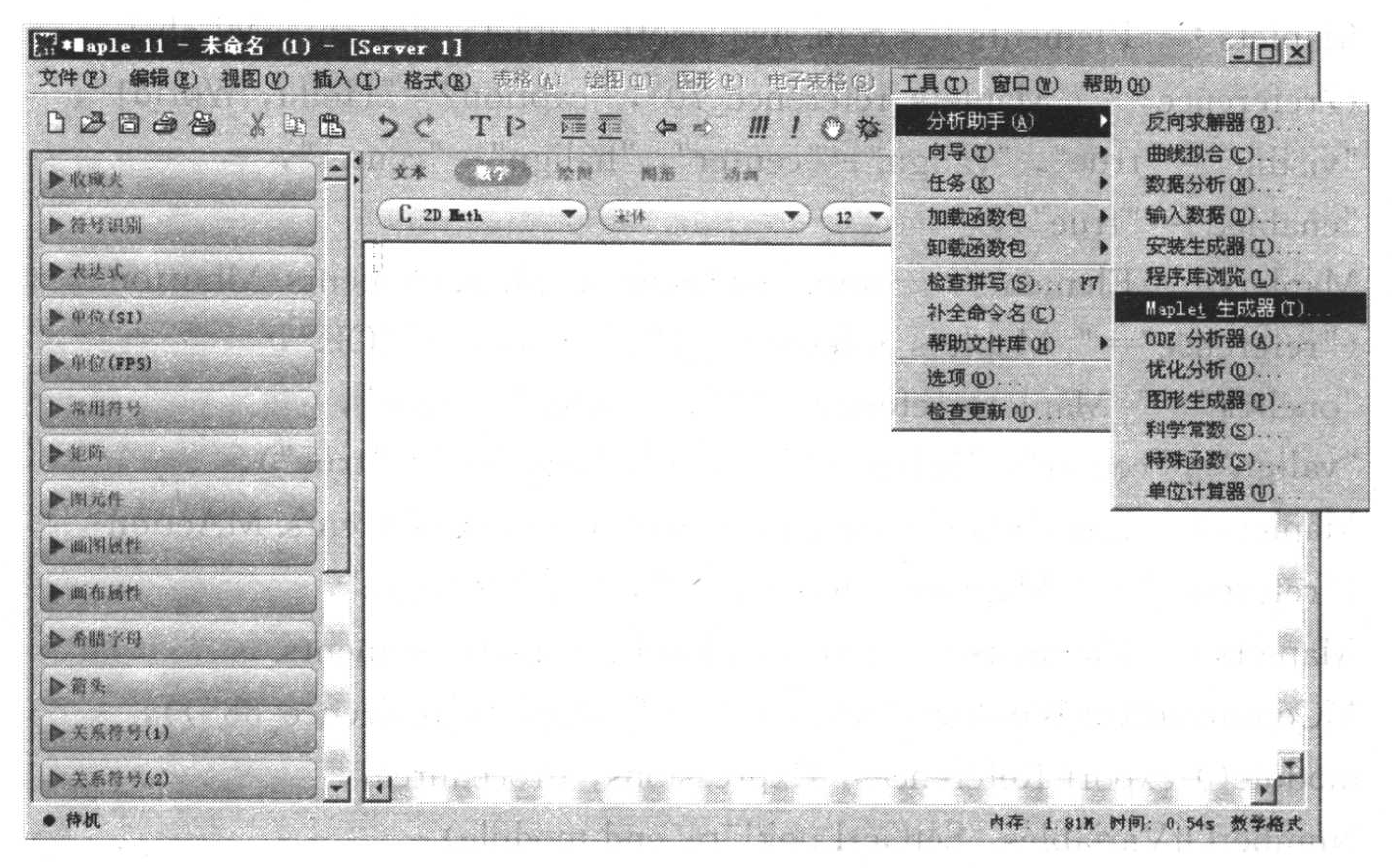


图 7-4

利用 Maplet Builder，我们可以设计 Maplet 的布局，通过鼠标的拖放将左侧工具栏中的菜单、按钮、文本框等 Windows 元素添加到 Maplet 中，并在右侧的属性栏中设置各元素的属性和关联的程序(图 7-5)。

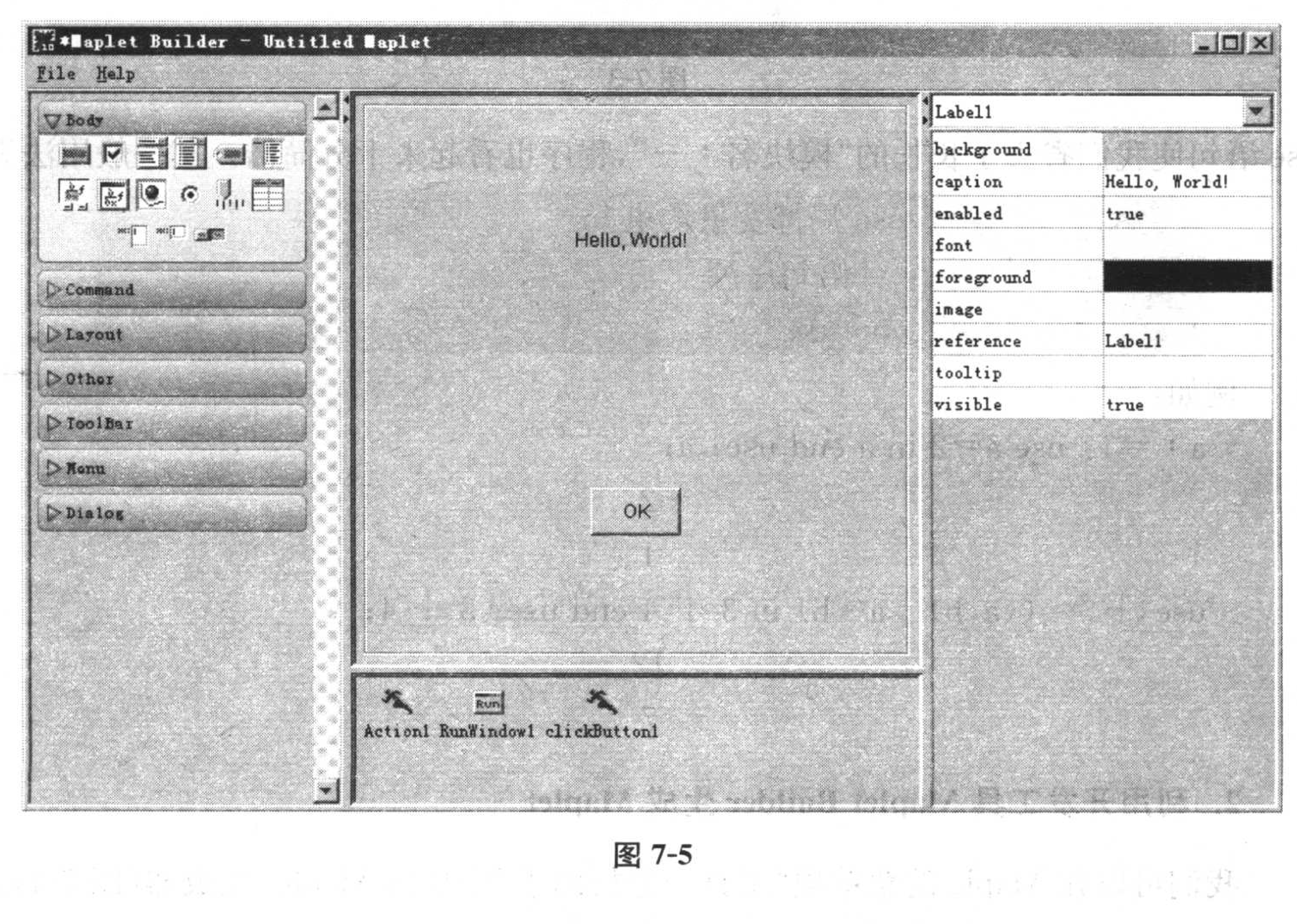


图 7-5

然后，点击 Maplet Buider 的菜单项 File-> Run 来运行 Maplet(图 7-6)。



图 7-6

7.6　程 序 调 试

Maple 提供了多种方式来方便我们对程序进行调试。我们既可以在编写程序的时候插入各种调试语句，也可以使用调试工具来调试正在运行的程序。例如：假设我们在例 2 中忘记了使用 break 语句，

```
> for x from 2 by 7 do
    if x mod 3=2 and  x mod 5=3 then print(x) end if
  end do;
```

程序将持续运行下去，直至系统崩溃。这时我们可以点击工具栏中的图标来中断程序的运行，或者点击图标来调试程序(图 7-7)。当然，在程序暂停之前我们需要等待很长一段时间，这时程序也已经打印出了几万行结果。

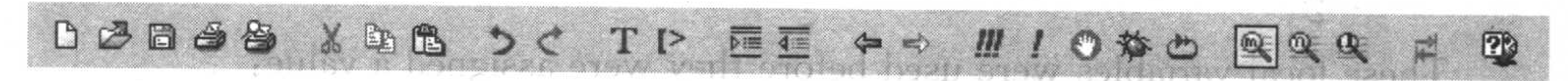


图 7-7

1. Mint

mint 是 Maple 的一个语法检查工具，既可以在 DOS/Linux 命令行下使用(mint)，也可以在 Windows 图形界面下使用(wmint)，还可以在 Maple 程序中使用(maplemint 语句)。mint 对包含 Maple 源程序的文本文件进行语法分析，报告其中可能的错误或者给出警告信息。mint 的命令行用法为：

```
mint [-klpqsSxv][-a…][-A…][-b…][-d…][-D…]
[-i…][-I…][-o…][-t…][-U…][-w…] 文件
```

maplemint 语句的用法为：

```
maplemint(过程)
```

例如：

```
> i：=-1; j：=-2; s：=-3; x：=-5;
> f：= proc(x)
      local i,j,s,t;
      for i to x do
         s：=0; for i to x do s：=s+i end do; t：=t+s;
      end do;
      return s;
      t：=0
   end proc;
> maplemint(f);
This code is unreachable:
t：= 0
These parameters have the same name as constants:
      -5
These local variables have the same name as constants:
      -3, -2, -1
These local variables were never used:
      -2
These local variables were used before they were assigned a value:
      t
These variables were used as the same loop variable for nested loops:
      -1
```

2. Maple Debugger

Maple Debugger 是 Maple 的一个图形化的调试工具。它提供一个窗口，我们可以在其中方便地输入调试命令看到结果，而不必担心源程序被淹没在调试信息之中。Maple Debugger 窗口（图 7-8）平时并不出现，只有当程序运行时我们点击了图标 ，程序遇到了断点（breakpoint）、监视点（watchpoint），或者我们执行了 DEBUG

语句的时候才会出现。

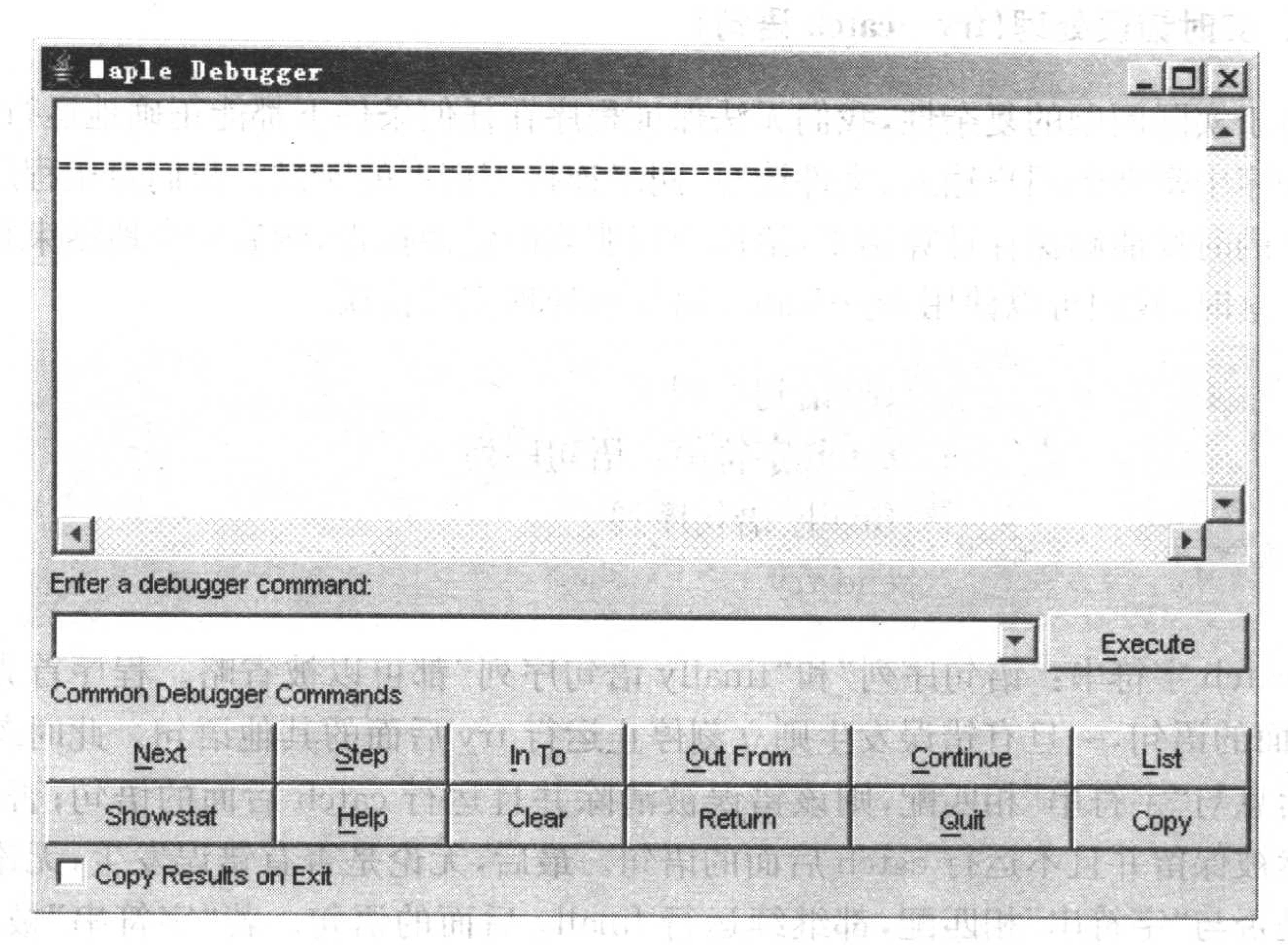


图 7-8

窗口中的 12 个调试命令按钮的作用见表 7-1。

表 7-1　Maple Debugger 的调试命令

Next	执行下一条语句。如果该语句具有嵌套结构或包含子程序调用，则该语句被作为一个整体来执行
Step	执行嵌套结构或子程序调用中的下一条语句
In To	执行嵌套结构(但不是子程序调用)中的下一条语句
Out From	继续程序运行，执行完当前语句或嵌套结构
Continue	继续程序运行，直至遇到新的断点、监视点
List	列出当前过程及当前语句
Showstat	显示所有相关过程的定义语句
Help	显示 Maple Debugger 的帮助
Clear	清除输出窗口的所有内容
Return	继续程序运行，从当前子程序中返回
Quit	结束程序调试，并关闭 Maple Debugger 窗口
Copy	将输出窗口的内容复制到当前编程区

3. 实时错误处理(try…catch 语句)

由于实际问题的复杂性,我们无法保证程序在任何条件下都能正确地运行,尤其是当程序中牵涉到用户输入、文件读写、网络连接等情形的时候。我们希望程序在错误发生的时候能够保存计算结果,给用户以明确的错误报告,然后安全地结束程序的运行。这时,我们可以使用 try…catch 语句来处理实时错误。

```
try 语句序列
catch 字符串：语句序列
finally 语句序列
end try
```

其中“catch 字符串：语句序列”和“finally 语句序列”都可以被省略。程序首先运行 try 后面的语句,一旦有错误发生则立刻停止运行 try 后面的其他语句。此时若某个错误信息与“字符串”相匹配,则该错误被清除并且运行 catch 后面的语句;否则,错误仍然被保留并且不运行 catch 后面的语句。最后,无论是否有错误发生,无论错误信息是否与“字符串”相匹配,都继续运行 finally 后面的语句。若“字符串”被省略,则认为与任何错误信息都匹配。例如:

```
> f := proc(x,y)
    local z;
    try z :=x/y
    catch "numeric exception":
        if x> 0 then z :=infinity
        elif x<0 then z :=-infinity
        else z :=unknown end if
    finally z
    end try
  end proc;
> f(-1,0),f(0,0),f(1,0);
```

$$-\infty,\ \text{unknown},\ \infty$$

并非所有实时错误 Maple 都可以处理,Maple 无法处理下列错误:用户中断,声明失败,堆栈溢出,对象太大,内存不足。

4. 调试语句(见表 7-2)

表 7-2　Maple 的调试语句

用　　法	功　　能
ASSERT(表达式,错误信息)	检验逻辑型"表达式"应该为真值。否则,显示"错误信息"。"错误信息"可省略,缺省值为空。在使用 ASSERT 语句之前,我们必须首先设置 Maple 的内核选项 kernelopts(ASSERT=true),否则 ASSERT 语句不起作用
DEBUG(表达式)	设置断点,同时在 Debugger 窗口中显示"表达式"的值。"表达式"可省略,缺省值为空
debugopts(语句序列)	设置 Maple 的各种调试选项
error 错误信息,参数列表	显示错误信息并立即中断程序的运行。"参数列表"可省略,缺省值为空
printlevel := 整数值	设置调试的深度,缺省值为 1。值越大,显示调试信息越多
showstat(过程,语句范围)	显示过程的语句及编号,断点的位置和当前语句的位置。"过程"可省略,缺省值为所有具有断点的过程。"语句范围"也可省略,缺省值为所有语句。"语句范围"之外的语句将不被显示出来,而是用"…"代替
showstop()	显示所有具有断点的过程、所有的断点、所有的监视点和所有的错误监视点
stopat(过程,语句号,表达式)	在"过程"中某语句处设置断点,并返回所有具有断点的过程列表。当逻辑型"表达式"为真值时,启动 Debugger。"语句号"可省略,缺省值为 1;"表达式"也可省略,缺省值为 true
stoperror(字符串)	设置与"字符串"相匹配的错误监视点,并返回所有错误监视点列表。当错误被发现时,启动 Debugger。若"字符串"被省略,则仅返回所有错误监视点列表
stoplast(语句号)	在发生上次错误的过程中某语句处设置断点。"语句号"可省略,缺省值为 1
stopwhen(变量)	设置关于"变量"的监视点,并返回所有监视点列表。当"变量"值发生变化时,启动 Debugger。若"变量"被省略,则仅返回所有监视点列表

续表

用　　法	功　　能
debug(过程列表) trace(过程列表)	追踪“过程列表”中各过程的运行情况，显示各过程语句的结果
tracelast	显示上次错误发生时的程序调用堆栈
unstopat(过程，语句号)	清除“过程”中某语句处的断点，并返回所有具有断点的过程列表。若“语句号”被省略，则清除“过程”中的所有断点。若“过程”被省略，则清除所有过程中的所有断点
unstoperror(字符串)	清除与“字符串”相匹配的错误监视点，并返回所有错误监视点列表。若“字符串”被省略，则清除所有错误监视点
unstopwhen(变量)	清除关于“变量”的监视点，并返回所有监视点列表。若“变量”被省略，则清除所有监视点
undebug(过程列表) untrace(过程列表)	停止对“过程列表”中各过程的追踪
WARNING(警告信息，参数列表)	显示警告信息
where(n)	显示程序调用堆栈最内的 n 层。n 可省略，缺省值为 1

虽然 Maple 具有强大的程序调试功能，但只有加强对编程语言的熟练掌握程度和养成良好的编程习惯，才可以使我们尽量减少错误的发生，写出高质量的程序。

习　　题

1. 输出 100 至 1000 之间的能被 3 或 11 整除的所有自然数。
2. 输出 1000 以内的水仙花数，即该数等于各位数字(十进制)的立方和。
3. 输出 1000 以内的完全数，即该数等于所有真因子之和。
4. 定义函数，计算并画出输入函数的一阶导数、二阶导数。
5. 定义矩阵 $\begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ 1 & 1-x & 1 & \cdots & 1 \\ 1 & 1 & 2-x & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & 1 & \cdots & n-x \end{pmatrix}$。

6. 定义矩阵 $\begin{pmatrix} 0 & 1 & 2 & \cdots & n-1 \\ 1 & 0 & 1 & \ddots & \vdots \\ 1 & 2 & 0 & \ddots & 2 \\ \vdots & \vdots & \ddots & \ddots & 1 \\ 1 & 2 & \cdots & n-1 & 0 \end{pmatrix}$。

7. 随机形成 100 以内的 n 维整数数组(a_i)，并输出该数组的算术平均 $\dfrac{a_1+a_2+\cdots+a_n}{n}$、几何平均 $\sqrt[n]{a_1a_2\cdots a_n}$ 和调和平均 $\dfrac{n}{\dfrac{1}{a_1}+\dfrac{1}{a_2}+\cdots+\dfrac{1}{a_n}}$。

8. 定义分段函数 $g(x)=\begin{cases} \log x, & x>10 \\ \mathrm{e}^x+1, & -10\leqslant x\leqslant 10 \\ |x|, & x<-10 \end{cases}$，并计算 $g(15)$，$g(5.2)$，$g(-15)$。

9. 求数列 $x_n=1+\dfrac{1}{2^3}+\cdots+\dfrac{1}{n^3}$ 极限的近似值，当 $|x_n-x_{n+1}|<10^{-5}$ 时停止。

10. 用弦截法 $x_k=x_{k-1}-\dfrac{(x_{k-1}-x_{k-2})f(x_{k-1})}{f(x_{k-1})-f(x_{k-2})}$，$x_0=-1$，$x_1=1$，求方程 $f(x)=x^3-2x^2+7x+4=0$ 根的近似解，要求 $|x_k-x_{k-1}|<10^{16}$。

11. 用 Newton 迭代公式 $x_{n+1}=\dfrac{1}{2}\left(x_n+\dfrac{a}{x_n}\right)$，计算正实数 a 的平方根。

12. 计算实矩阵 A 的范数 $\|A\|=\sqrt{\lambda_1(A^{\mathrm{T}}A)}$，即 $A^{\mathrm{T}}A$ 的最大特征根的平方根。

13. 计算曲线 L：$\begin{cases} x(t)=a\cos^3 t \\ y(t)=a\sin^3 t \end{cases}$，$0\leqslant t\leqslant 2\pi$ 所围区域的面积 $A=\dfrac{1}{2}\int_L x\mathrm{d}y-y\mathrm{d}x$。

14. 输入实对称阵 A，输出对角矩阵 D 和正交矩阵 Q，满足 $D=Q^{\mathrm{T}}AQ$。

15. 输出如下形式的 n 阶方阵 A。例如，$n=5$，$A=\begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 16 & 17 & 18 & 19 & 6 \\ 15 & 24 & 25 & 20 & 7 \\ 14 & 23 & 22 & 21 & 8 \\ 13 & 12 & 11 & 10 & 9 \end{pmatrix}$。

第 8 章　Maple 操作命令

典型的 Maple 11 文件模式窗口(图 8-1)由主菜单、工具栏、输入面板、工作窗口、上下文工具栏、状态栏 6 部分组成。在本例中,工作窗口含有 3 个工作表“未命名(1)”,“未命名(2)”,“UserManual,Contents”。状态栏用来显示诊断信息、Maple 系统内核所占用的内存和处理器时间等。

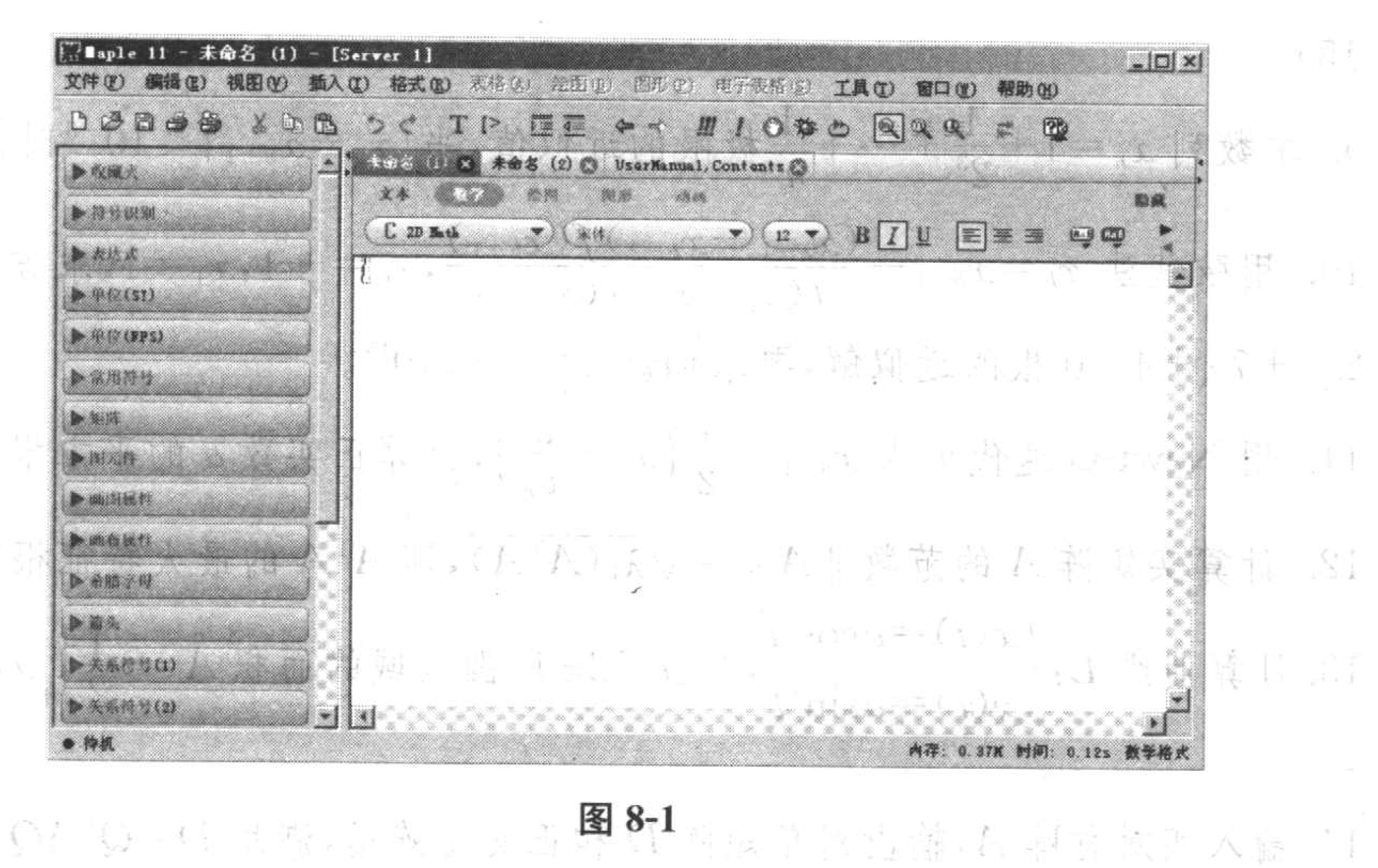


图 8-1

8.1　菜 单 命 令

Maple 11 共有 12 个主菜单:文件、编辑、视图、插入、格式、表格、绘图、图形、电子表格、工具、窗口、帮助。其中只有当光标位于相应的对象上时,表格、绘图、图形、电子表格菜单才会处于激活状态,即菜单命令为深色,否则菜单命令为浅色的无效状态。每个主菜单还有若干菜单项和子菜单项。

在工作窗口的某一输入对象的位置处单击鼠标右键,会弹出上下文的快捷菜单。此处不一一赘述,读者可在使用中慢慢体会。

8.2 工　具　栏

Maple 11 的工具栏中共有 26 个命令。将光标置于命令的图标处时，我们可以看到有关该命令功能的屏幕提示。

8.3 上下文工具栏

Maple 11 共有 4 种上下文工具栏：文本/数学、绘图、图形、动画。当光标位于不同的对象上时，出现相应的上下文工具栏。

8.4 输 入 面 板

Maple 11 共有 30 组输入面板，其中 8 个为字母表，9 个用于输入 Maple 表达式，11 个用于输入数学符号，还有 2 个用于绘图。通常只显示缺省面板。可通过鼠标右击输入面板更改设置(图 8-2)。



图 8-2

8.5 获取帮助示例

【例 1】 用“Maple T. A”学习中值定理。

运行 Maple 11，进入文件模式窗口。

点击菜单项“帮助→Maple 漫游”，显示“The Maple Tour”(图 8-3)。

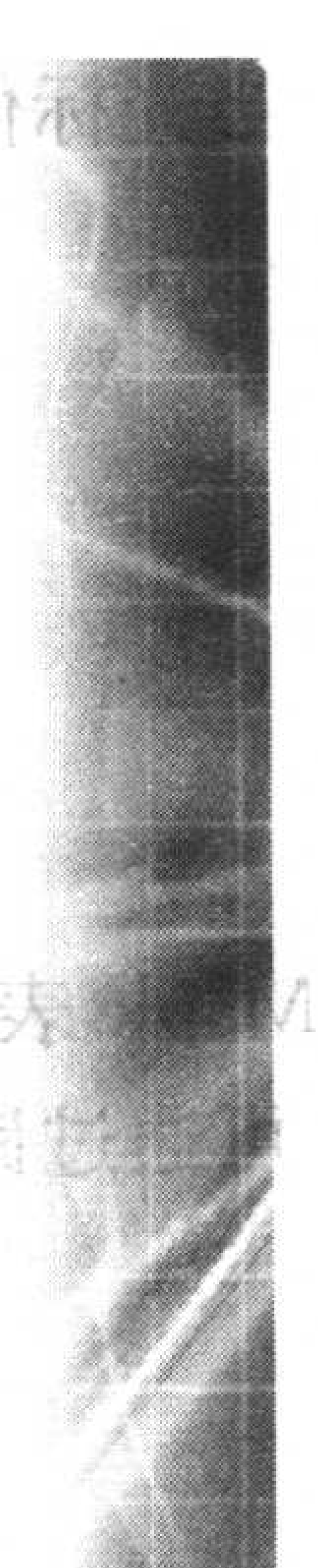


图 8-3

点击链接Ten Minute Tour，跟随帮助浏览，不到 10 分钟你就会了解 Maple 的轮廓。

点击链接Education Assessment Maple T. A.，调出 Maple 的辅助教学系统(图 8-4)。

▾Education, Assessment, Maple T.A.

The Student packages allow instructors to effectively deliver course content and they allow students to gain insight, practice as well as deepen their understanding. Student packages are available for the following topics:

Pre-Calculus	Calculus
Multivariate Calculus	Vector Calculus
Linear Algebra	

Maple also allows you to deliver assignments and tests right inside of a Maple worksheet. Students are given instant feedback on their level of understanding. Assignments authored within Maple can also be exported to an online Maple T.A. system. Click below for more information:

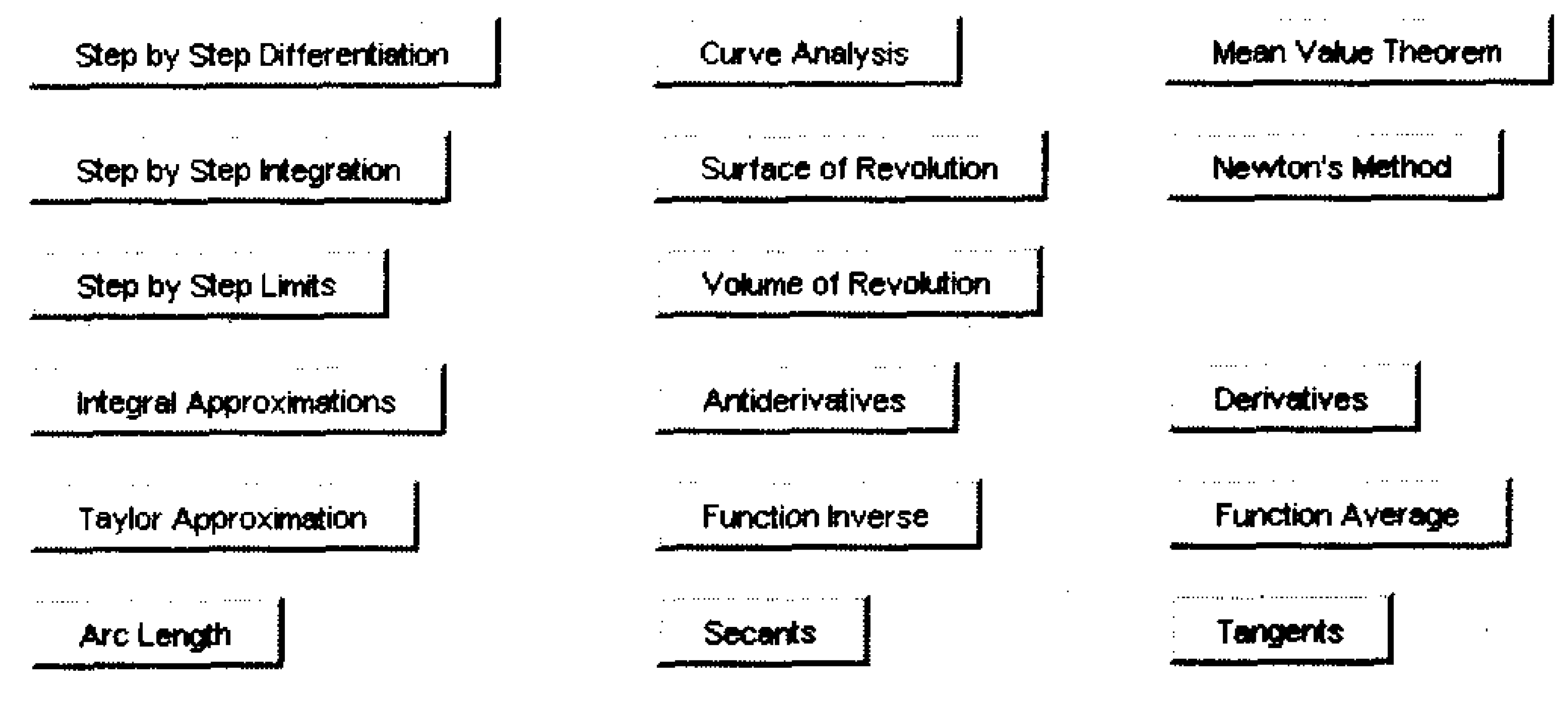
Assignments in Maple

图 8-4

点击链接Calculus，调出关于单变量微积分的课件(图 8-5)。

▾Single-variable Calculus

The Student[Calculus1] package includes a number of interactive tutors that address specific subjects in a (single-variable) Calculus course. Click the buttons below to launch examples of these tutors. They can also be found on the **Tools>Tutors** menu.



These tutors can be used by an instructor to illustrate a concept in class. The tutors are also useful for the students to review and practice on their own.

图 8-5

点击 Mean Value Theorem 按钮，运行中值定理课件。在输入函数 $f(x)=$ sin(x)+x^3，区间 $a=0$，$b=1.2$ 后，点击 Display 按钮，给出满足 $f'(c)=\dfrac{f(b)-f(a)}{b-a}$ 的点 $c=0.695$(图 8-6)。

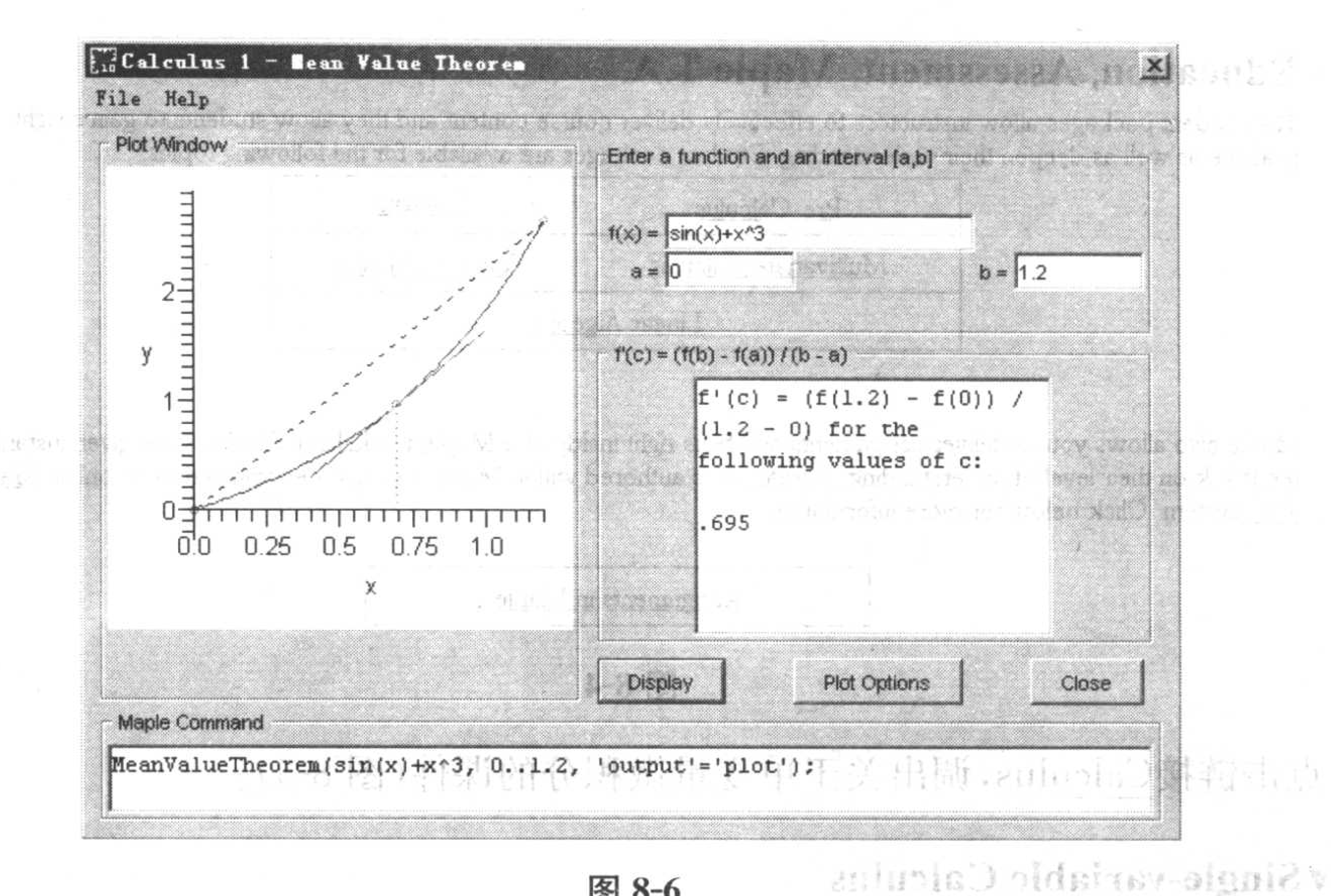


图 8-6

菜单项“工具→向导→微积分一单变量→平均值定理”也同样可以显示如上图所示的对话框。

【例 2】 用菜单命令观察函数的泰勒展开。

运行菜单项“工具→向导→微积分一单变量→泰勒近似”。在输入函数 $f(x)=x*\cos(x)$，初值点 $x=0$，展开次数 Order=4 后，点击 Display 按钮，在同一个坐标系中显示函数及其泰勒展开的图像(图 8-7)。

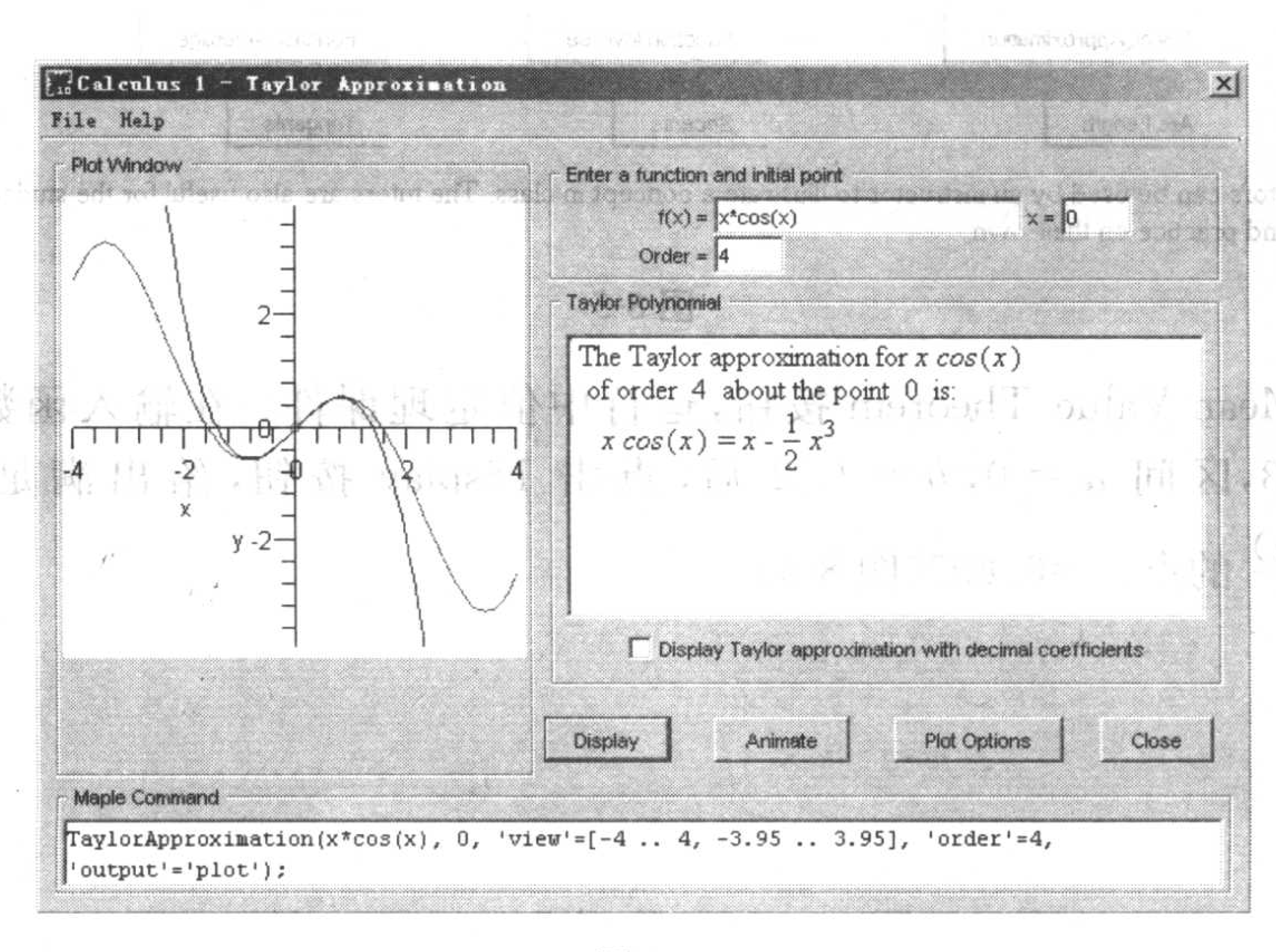


图 8-7

【例 3】 调出有关矩阵运算命令。

用鼠标右键点击矩阵，则可以调出许多有关矩阵运算的函数。常用的有：

· 标准函数（图 8-8）：维数 Dimensions、秩 Rank、行列式 Determinant、逆矩阵 Inverse、范数 Norm、迹 Trace、转置矩阵 Transpose。

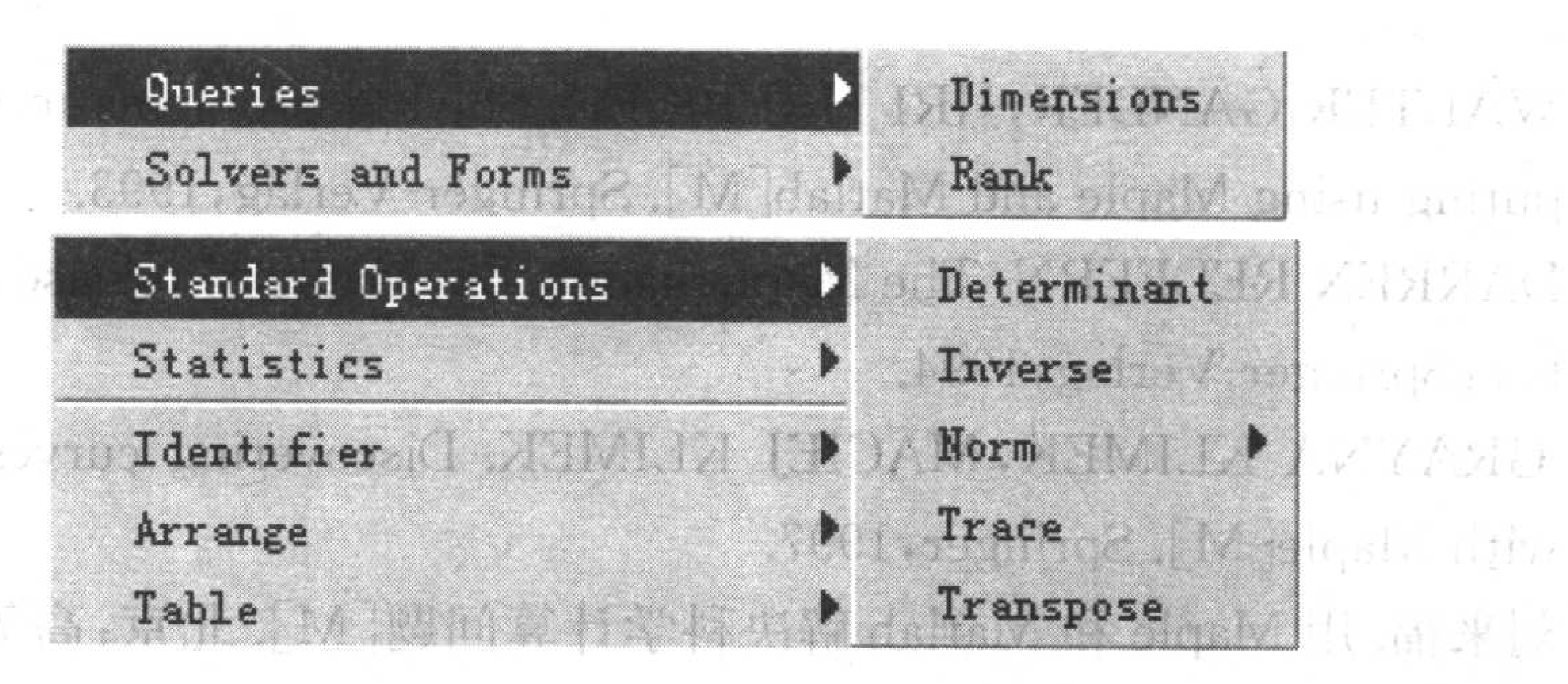


图 8-8

· 特征值相关的函数（图 8-9）：特征多项式 Characteristic Polynomial、特征值 Eigenvalues、特征向量 Eigenvectors、奇异值 Singular Values。

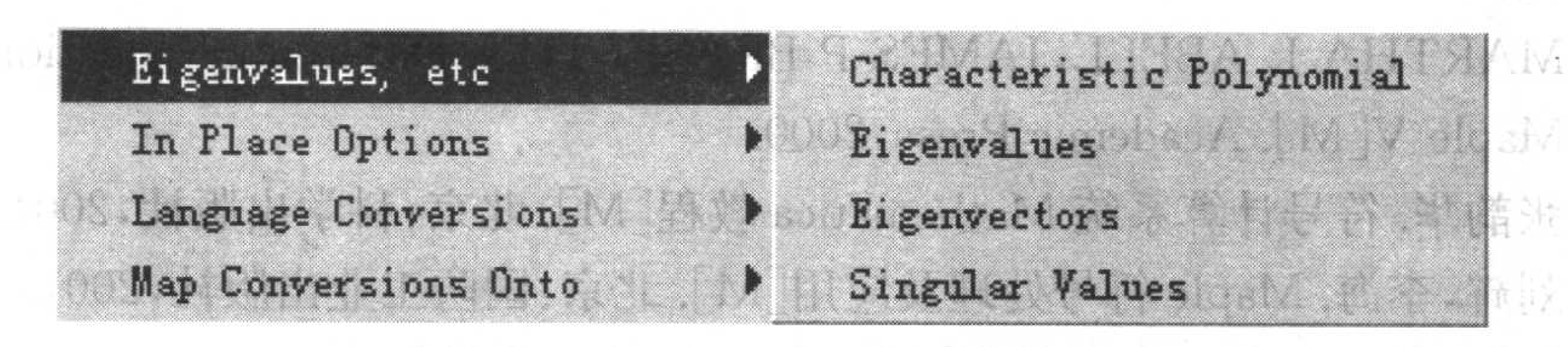


图 8-9

· 矩阵分解和标准形（图 8-10）：Cholesky 分解、LU 分解、QR 分解、Frobenius 标准形、Hermite 标准形、Jordan 标准形、Smith 标准形。

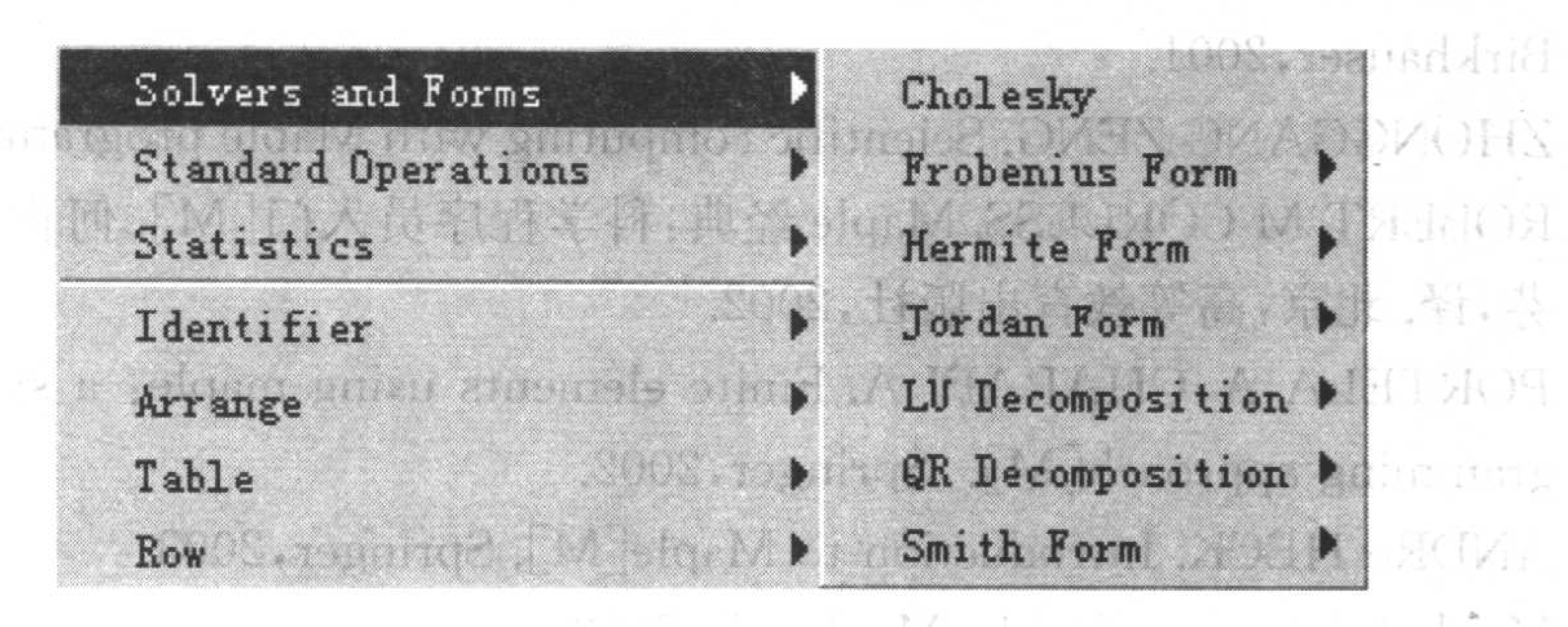


图 8-10

参 考 文 献

[1] WALTER GANDER, JIRI HREBICEK. Solving problems in scientific computing using Maple and Matlab[M]. Springer-Verlag, 1993.

[2] DARREN REDFERN. The Maple handbook: Maple V release 3[M]. Heidelber: Springer-Verlag, 1994.

[3] GRAYNA KLIMEK, MACIEJ KLIMEK. Discovering curves and surfaces with Maple[M]. Springer, 1997.

[4] 刘来福. 用 Maple 和 Matlab 解决科学计算问题[M]. 北京:高等教育出版社, 1999.

[5] 马春庭. 掌握和精通 Maple[M]. 北京:机械工业出版社, 2000.

[6] ERNIC KAMERICH. Maple 指南[M]. 唐兢, 李静, 译. 北京:高等教育出版社, 2000.

[7] MARTHA L ABELL, JAMES P BRASELTON. Differential equations with Maple V[M]. Academic Press, 2000.

[8] 张韵华. 符号计算系统 Mathematica 教程[M]. 北京:科学出版社, 2001.

[9] 刘辉, 李海. Maple 符号处理及应用[M]. 北京:国防工业出版社, 2001.

[10] 洪伟, 等. Maple 6 实用教程[M]. 北京:国防工业出版社, 2001.

[11] ELIAS DEEBA, ANANDA GUNAWARDENA. 用 Maple V 学习线性代数[M]. 丘维声, 译. 北京:高等教育出版社, 2001.

[12] STEPHEN LYNCH. Dynamical systems with applications using Maple[M]. Birkhauser, 2001.

[13] ZHONGGANG ZENG. Scientific computing with Maple programming. 2001。

[14] ROBERT M CORLESS. Maple 经典:科学程序员入门[M]. 何青, 袁荣, 王丽芬, 译. 北京:高等教育出版社, 2002.

[15] PORTELA A, CHARAFI A. Finite elements using maple: a symbolic programming approach[M]. Springer, 2002.

[16] ANDRé HECK. Introduction to Maple[M]. Springer, 2003.

[17] Maple 9 Learning Guide. Maplesoft. 2003.

[18] MONAGAN M B, GEDDES K O, HEAL K M, et al. Maple 9 introductory programming guide. Maplesoft. 2003.

[19] MONAGAN M B, GEDDES K O, HEAL K M, et al. Maple 9 advanced

programming guide. Maplesoft. 2003.

[20] 黎捷. Maple 9.0 符号处理及应用[M]. 北京:科学出版社,2004.

[21] 李尚志,陈发来,张韵华,等. 数学实验[M]. 北京:高等教育出版社,2004.

[22] 张晓丹,等. Maple 的图形动画技术[M]. 北京:北京航空航天大学出版社,2005.

[23] Martha L Abell, James P Braselton. Maple by example[M]. Elsevier Academic Press, 2005.

[General Information]
书名=符号计算系统MAPLE教程
作者=张韵华　　王新茂编著
页数=221
SS号=11911963
出版日期=2007年10月第1版

参考文献